REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE 3.Jan.00	3. REPORT TYPE AND DATES COVERED THESIS

4. TITLE AND SUBTITLE A COMPARISON TO TWO MENTODS USED FOR RANKING TASK EXPOSURE LEVELS USING SIMULATED DATA	5. FUNDING NUMBERS
6. AUTHOR(S) CAPT COSTANTINO JOSEPH	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) UNIVERSITY OF OKLAHOMA HSC	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) THE DEPARTMENT OF THE AIR FORCE AFIT/CIA, BLDG 125 2950 P STREET WPAFB OH 45433	10. SPONSORING/MONITORING AGENCY REPORT NUMBER FY99-665
11. SUPPLEMENTARY NOTES	
12a. DISTRIBUTION AVAILABILITY STATEMENT Unlimited distribution In Accordance With AFI 35-205/AFIT Sup 1	12b. DISTRIBUTION CODE
13. ABSTRACT *(Maximum 200 words)*	
14. SUBJECT TERMS	15. NUMBER OF PAGES 85
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

R 7

THE UNIVERSITY OF OKLAHOMA HEALTH SCIENCES CENTER

GRADUATE COLLEGE

A COMPARISON OF TWO METHODS USED FOR RANKING TASK EXPOSURE LEVELS USING SIMULATED MULTI-TASK DATA

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

degree of

Master of Science

BY

JOSEPH COSTANTINO

Oklahoma City, Oklahoma

1999

2000 112 038

A COMPARISON OF TWO METHODS USED

FOR RANKING TASK EXPOSURE LEVELS

USING SIMULATED MULTI-TASK DATA

APPROVED BY:

[illegible]
Margaret L. Phillips, Ph.D., CIH, Chair

[illegible]
Nurtan A. Esmen, Ph.D.

[illegible]
David L. Johnson, Ph.D., CIH

Thesis Committee

TABLE OF CONTENTS

LIST OF TABLES

CHAPTER I

INTRODUCTION AND LITERATURE REVIEW

The role of the occupational hygienist is to protect the health and well-being of workers and the public through anticipation, recognition, evaluation, and control of hazards arising in or from the workplace.[1] During the evaluation phase, an occupational hygienist may collect air samples to quantify workers' exposure levels. Documentation of exposure-related factors, or determinants, is an important aspect of both the sampling event and the comprehensive exposure evaluation. Some examples of determinants include: worker location, raw materials and equipment used, engineering controls, environmental conditions, and task. Researchers have studied determinants of exposure to identify factors that are associated with an increase or decrease in exposure levels.[2] Determinants are observed and recorded during experimental and observational studies.

In most cases, experimental studies are designed to evaluate the effectiveness of engineering controls.[3-6] For example, studies have assessed the effect of various ventilation configurations on oxygen levels in confined spaces,[3] exposure levels of worker's handling a flour additive powder,[4] and exposure to a pesticide surrogate during spraying in a greenhouse.[5]

Observational studies attempt to identify the effect of various determinants on exposure levels under actual working conditions. For jobs consisting of a variety of tasks at different locations, the occupational hygienist may find it useful to identify those tasks with high exposure levels. The identification of these tasks allows the hygienist to take preventive measures to reduce these exposure levels through the implementation of engineering controls, changes in work practices, or personal protective equipment.

Several sampling strategies have been used to identify determinants of exposure such as: area sampling, full-shift sampling, and task-specific sampling. In an ideal situation, the occupational hygienist would conduct an exposure assessment for each individual task to rank them according to exposure levels; however, this is not cost effective, nor does it allow for use of existing multi-task exposure assessment data. Many studies have used a task-specific sampling strategy.[7-10] Task sampling was conducted during a highway construction project in an attempt to establish baseline exposure levels to such tasks as: digging trenches, paving asphalt, and grinding road cover.[7] The study used the project budget to define construction stages (i.e., earthworks, drainage) and then the tasks within each stage were identified. Exposures to noise, dust, and asphalt fumes were measured on operating engineers and laborers. A printing plant study[8] evaluated the influence of task and duration on solvent exposures. Instantaneous samples were collected during tasks performed by seven offset press operators over a three day period. Maximum solvent concentrations were measured during a plate change task. One advantage of the task-specific strategy is that peak exposures may be identified, particularly if direct-reading instrumentation is used.

The area sampling approach places monitoring equipment close to the sources of concern.[11-13] Area samples of antineoplastic agents were collected in an outpatient oncology clinic and pharmacy.[11] In these two areas antineoplastic agents were prepared and administered. Personal sampling was not feasible because of the large air sample volumes required to meet minimum detection limits. To quantify ambient concentrations of dust generated from wood-working machines, area samples were collected in three factories.[12] The area sampling results identified the cross-cut saw, horizontal belt sander, and the plate saw as the machines generating maximum concentrations in each respective factory. Studies using area sampling only are limited because personal exposures are usually underestimated and the worker's behavior cannot be evaluated.[2]

In practice, much industrial hygiene sampling has been compliance driven. The compliance sampling strategy usually uses worst-case monitoring with a focus on exposures during the time of the survey. An attempt is made to identify the maximum-exposed workers in a group. One or a few measurements are taken and simply compared with the occupational exposure limit.[14] A comprehensive exposure assessment strategy would include an evaluation of all potential hazards so that exposure levels are characterized for all workers, on all days. This strategy is not used very often due to the high cost of collecting and analyzing so many samples. The compliance strategy is reactive while the comprehensive strategy is proactive. Current emphasis in exposure assessment is focused on moving beyond "compliance management" to "risk management." Since most occupational exposure limits are full-shift time-weighted averages (TWAs),[15] existing exposure assessment data consists largely of full-shift or partial-shift samples, often spanning multiple tasks.

Four studies have attempted to identify high exposure tasks using multi-task sampling data.[16-19] These studies constructed multiple linear regression models to estimate the relationships between the tasks performed and measured exposure levels. Multiple regression analysis is a mathematical technique used to determine the relationships between a dependent variable and multiple independent variables.[20] However, the results of the data analyses in these studies were not validated by comparison with simultaneous single-task sampling results. Therefore, there is no way to know if these models were accurate.

The objectives of a bakery study[16] were to measure full-shift exposure to inhalable dust in bakeries and define the determinants of full-shift exposure. The study used a cross-sectional design with one exposure measurement from each individual in the recruited bakeries. Ninety-six workers, employed in seven different bakeries, participated in the study. Two side-by-side full-shift inhalable dust samples were obtained from each study participant. Multiple linear regression was used to identify the combination of independent variables that had the best ability to explain full-shift inhalable dust exposure levels. The multiple regression model was tested for violation of the assumptions of regression analysis. Assumptions of homoscedasity and linearity were tested via graphical methods. The model indicated which tasks were associated with increasing or decreasing exposure levels. The tasks which the model predicted to be associated with increasing exposures included: dough-forming, bread and bun production, and flour pouring and dusting.

The main objective of a pig farm study[17] was to use modeling to obtain a more valid measure of long-term average exposure for epidemiologic purposes. The model was constructed using a limited number of measurements and by using surrogate measures of exposure. This study suggested that most long-term average exposure estimates are imprecise due to intraindividual variability and a limited number of measurements. In a group of 198 Dutch pig farmers, exposure to endotoxins was measured on one workday in the summer and one workday in the winter. In the summer and winter, the farmers were requested to complete a diary on time spent in different activities during the day of the exposure measurement and the following six days. Time spent in each activity was recorded. In a subgroup of six farmers, exposure measurements were performed nearly monthly during a one-year period. Farm characteristics such as number of animals, feeding methods, heating and ventilation, type of floor, and bedding material were recorded during walk-through surveys. The data set contained 95 distinct variables. The multiple linear regression analysis identified those tasks (ear tagging, teeth cutting) which were associated with increased exposure levels.

A rubber manufacturing industry study[18] assessed chemical exposures in ten plants. Personal exposures to airborne particulates, rubber fumes and solvents, and dermal contamination were measured. Information on tasks performed, ventilation characteristics, and production variables were used in multiple regression models to identify those factors which affected exposure levels. Model adequacy was tested with standard regression techniques such as residual plots and outlier detection. The multiple regression analysis predicted which tasks (cleaning, weighing, jointing) were statistically significantly associated with higher exposure levels.

A lumber mill study quantified metals exposures of saw filers.[19] Observations of tasks, locations, and activities were recorded in ten minute intervals. A stepwise multiple linear regression model was used to identify the determinants of exposure. Maximum exposure levels to cobalt and chromium were associated with wet carbide grinding and knife grinding, respectively.

An alternative method to rank task exposures using time-weighted average (TWA) samples has recently been studied.[21] This method, referred to as the P-screen method, was evaluated using simulated data. (See Materials and Methods for details on data simulation and the P-screen methodology.) The P-screen method ranked the two highest exposure tasks correctly 100% of the time, if the number of samples was adequate and the task distributions were not highly overlapped. The performance of the model improved with decreasing task distribution geometric standard deviation (GSD), increased spacing of the task distributions, and an increase in the number of samples. The model proved to be most useful for stratifying exposure levels into high, medium, and low categories.

Several other data analysis methods have been used to identify determinants of exposure such as: arithmetic means, geometric means, Analysis of Variance, and Kruskal-Wallis.[2] However, none of these methods can rank tasks using multi-task data with limited information on task times. Only the P-screen method and the multiple linear regression method have this capability. Therefore, it is appropriate to compare the performance of these two methods. The focus of this study was to conduct a side-by-side comparison of these two methods using simulated multi-task data.

CHAPTER II

PURPOSE AND SCOPE

This study compared the performance of two methods, P-screen and multiple linear regression, at ranking task exposures. Monte Carlo methods using simulated data were used to assess performance of the methods under a variety of experimental conditions.

CHAPTER III

METHODS AND MATERIALS

The existing computer program used for the P-screen method study[21] was modified to incorporate the multiple linear regression analysis method. The program was written in Microsoft QuickBASIC (see Appendix B). The following parameters were controllable by the programmer: task GSDs, spacing between task median concentrations, and number of samples. The number of tasks was arbitrarily fixed at six. Subject to these parameters, task durations and task concentrations were created through random number generation. The simulated data consisted of multi-task TWA concentrations. This study evaluated how performance of the methods were affected by three experimental parameters: (1) number of samples (J = 20 or 100), (2) nominal GSD ($\sigma_g = 2 \pm 0.5$ or $\sigma_g = 4 \pm 0.5$), and (3) overlap between distributions (20 - 80 % overlap). The overlap parameter was determined by the GSD and the task medians; two different task median spacings were already integrated in the program so that each experiment evaluated the performance for two sets of task distributions. The task distributions were log-normally distributed. The Type 1 task distribution medians were separated by a factor of two with actual task medians equal to 1, 2, 4, 8, 16, and 32, in arbitrary units. The Type 2 task distribution medians were separated by a factor of $2^{1.5}$ with actual task medians equal to 0.5, 1.4, 4, 11.3, 32, and 90.5, in arbitrary units.

Generation of Simulated Data

The following is a summary of the steps to generate simulated TWA data.[21]

1. The task time matrix θ ("I" task times for each of "J" samples) was randomly generated. Thirty task time matrices were generated for each experiment. In each sample, at least one task time was randomly assigned a zero value (i.e., the task does not occur). All other task times θ_{ij} were assigned random times as a discrete fraction of the sampling time (1/96, 2/96, ..., 96/96), such that the sum of the task times was equal to one.
2. The task concentrations C_{ij} were created. These values were randomly selected from established task concentration distributions. One hundred task concentration matrices were generated for each task time matrix (30 task time matrices X 100 concentration matrices = 3000 trials per experiment).
3. Time-weighted averages C_j for each sample are calculated by $\sum_i \theta_{ij} C_{ij}$

P-screen Method

The following is a summary of the steps taken to estimate task median concentrations using the P-screen method.[21]

1. The natural logarithm of each time-weighted average sample concentration (C_j) was taken.

2. The P-screen matrix was set up for each sample where 0 = task performed during sample and 1 = task not performed during sample.

3. The P-screen matrix transpose was multiplied by log C_j to get the raw P-sum vector for each sample. This step aggregated TWA sample concentrations based on the non-occurrence of a particular task during that sample. The P-sum vector was normalized by dividing each element by the number of samples contributing to that element.

4. Task time weights were estimated for each sample by assigning $\hat{\theta}_{ij}$ equal to $1/m$ where m tasks were performed during the sample for at least 5% of the sample period. All other estimated task times are set equal to zero.

5. The P-screen matrix transpose for each sample was multiplied by the $\hat{\theta}_{ij}$ matrix to get the P-screened $\hat{\theta}$ matrix. The P-screened $\hat{\theta}$ matrix was normalized by dividing each row by the number of samples contributing to that row.

6. The normalized P-screened $\hat{\theta}$ matrix was inverted.

7. The normalized P-sum vector was multiplied by the inverted P-screened $\hat{\theta}$ matrix to get estimated task median concentrations (log c_i^{med}) for each sample.

8. Estimated task median concentration rankings were compared with actual rankings.

Multiple Linear Regression

The module written for the multiple linear regression produced regression coefficients for each task based on the model:

$$\ln c = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots \beta_t X_t \,. \tag{1}$$

where: c was the TWA sample concentration

β_0 was the intercept

β s were the regression coefficients

Xs were dichotomous variables representing the occurrence ($X_i = 1$) or nonoccurrence ($X_i = 0$) of the task

The intercept and regression coefficients were obtained by linear algebra methods using the following equations

$$\overleftrightarrow{LRA} \cdot \vec{X} = \overrightarrow{LRB} \tag{2}$$

$$\vec{X} = \overleftrightarrow{LRA}^{-1} \cdot \overrightarrow{LRB} \tag{3}$$

where the LRA matrix and the LRB vector were as defined in Appendix A and X was the vector of regression coefficients. The LRA matrix was set up based on the occurrence or nonoccurrence (<5% of the sample period) of a task. This matrix was only changed when a new task time matrix was generated. The LRB vector was set up based on the sample concentration and the task occurrence or nonoccurrence (see Appendix A for an example). The LRAB matrix (7x8) was composed of the LRA matrix (7x7) and the LRB vector (7x1). The LRAB matrix was sent to the MATSOLV1 subroutine for solution to obtain the regression coefficients. The magnitude and orientation of the regression coefficients were used to rank task concentrations. The regression coefficients were also used to estimate single-task median concentrations. As each module was incorporated into the existing program, a spreadsheet was used to validate randomly selected runs for at least two different task time matrices.

CHAPTER IV

RESULTS

A review of the summary statistics showed that the performance of the two methods was very similar. Summary statistics that were calculated included: (1) the probability of correctly ranking each task (Tables 1 - 8), (2) the number of runs that the highest two tasks were correctly ranked (Table 9), and (3) the number of correctly ranked runs (Table 10). As expected, model performance decreased as overlap and GSD increased and as the number of samples decreased.

A misclassification index (MI) was calculated to compare "how bad" the ranking was for each of the 3000 runs. Misclassification indices were computed for each data set (task time matrix) and the overall index for the entire experiment (Table 11). An MI of 0 indicated a correctly ranked run, while an MI of 1 indicated a worst-case ranking (i.e., reversed order 6, 5, 4, 3, 2, 1). The MI was calculated as follows:[22]

$$MI = \frac{\sum \left| actual\ rank - assigned\ rank \right|}{18} \qquad (4)$$

A One-way Analysis of Variance (ANOVA) of the MIs was conducted to test the null hypothesis that the MIs between the 30 task time matrices were all equal. All of the One-way ANOVAs resulted in a rejection of the null hypothesis. The results of this analysis

Table 1

Probability of Correctly Ranking Each Task:
Type 1 Distribution with J=20 and GSD=4.

Multiple Linear Regression

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.336**	0.264	0.189	0.123	0.070	0.018
	2	0.290	**0.267**	0.190	0.147	0.077	0.028
True	3	0.244	0.219	**0.210**	0.176	0.101	0.049
Rank	4	0.092	0.162	0.228	**0.266**	0.189	0.063
	5	0.027	0.063	0.123	0.184	**0.330**	0.273
	6	0.010	0.026	0.060	0.103	0.232	**0.569**

P-screen Method

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.360**	0.260	0.175	0.118	0.065	0.021
	2	0.264	**0.261**	0.205	0.145	0.084	0.040
True	3	0.227	0.230	**0.217**	0.173	0.099	0.054
Rank	4	0.105	0.152	0.204	**0.249**	0.203	0.087
	5	0.034	0.066	0.137	0.197	**0.301**	0.264
	6	0.010	0.030	0.061	0.118	0.247	**0.534**

Table 2

Probability of Correctly Ranking Each Task:
Type 2 Distribution with J=20 and GSD=4.

Multiple Linear Regression

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.320**	0.268	0.194	0.142	0.067	0.009
	2	0.301	**0.264**	0.212	0.137	0.069	0.017
True	3	0.269	0.235	**0.220**	0.165	0.081	0.031
Rank	4	0.089	0.171	0.235	**0.300**	0.176	0.029
	5	0.017	0.050	0.113	0.191	**0.413**	0.217
	6	0.004	0.012	0.026	0.065	0.195	**0.698**

P-screen Method

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.340**	0.273	0.185	0.131	0.058	0.013
	2	0.269	**0.271**	0.219	0.142	0.077	0.022
True	3	0.265	0.227	**0.226**	0.169	0.075	0.039
Rank	4	0.099	0.158	0.222	**0.271**	0.202	0.047
	5	0.023	0.055	0.120	0.203	**0.376**	0.222
	6	0.003	0.016	0.028	0.084	0.213	**0.657**

Table 3

Probability of Correctly Ranking Each Task:
Type 1 Distribution with J=20 and GSD=2.

Multiple Linear Regression

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.333**	0.296	0.181	0.141	0.048	0.001
	2	0.451	**0.212**	0.182	0.116	0.038	0.002
True	3	0.169	0.293	**0.295**	0.174	0.061	0.008
Rank	4	0.046	0.177	0.257	**0.345**	0.145	0.030
	5	0.002	0.019	0.063	0.160	**0.504**	0.253
	6	0.000	0.004	0.022	0.064	0.204	**0.706**

P-screen Method

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.356**	0.302	0.173	0.120	0.047	0.002
	2	0.426	**0.246**	0.158	0.126	0.039	0.005
True	3	0.173	0.287	**0.337**	0.159	0.036	0.008
Rank	4	0.044	0.141	0.250	**0.383**	0.146	0.036
	5	0.001	0.017	0.060	0.153	**0.524**	0.245
	6	0.001	0.007	0.023	0.058	0.207	**0.704**

Table 4

Probability of Correctly Ranking Each Task:
Type 2 Distribution with J=20 and GSD=2.

Multiple Linear Regression

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.307**	0.272	0.186	0.189	0.045	0.001
	2	0.427	**0.208**	0.192	0.147	0.025	0.000
True	3	0.209	0.300	**0.303**	0.150	0.035	0.003
Rank	4	0.056	0.210	0.261	**0.355**	0.108	0.010
	5	0.001	0.010	0.054	0.133	**0.624**	0.179
	6	0.000	0.000	0.005	0.026	0.163	**0.806**

P-screen Method

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.338**	0.287	0.173	0.156	0.045	0.001
	2	0.414	**0.226**	0.170	0.160	0.029	0.001
True	3	0.200	0.312	**0.325**	0.133	0.025	0.005
Rank	4	0.046	0.165	0.279	**0.376**	0.117	0.016
	5	0.001	0.008	0.047	0.142	**0.616**	0.186
	6	0.000	0.001	0.006	0.034	0.169	**0.791**

Table 5

Probability of Correctly Ranking Each Task:
Type 1 Distribution with J=100 and GSD=4.

Multiple Linear Regression

		Assigned Rank					
		1	2	3	4	5	6
True Rank	1	**0.615**	0.255	0.100	0.028	0.002	0.000
	2	0.254	**0.391**	0.264	0.081	0.010	0.001
	3	0.110	0.258	**0.380**	0.219	0.033	0.000
	4	0.021	0.092	0.228	**0.545**	0.110	0.003
	5	0.000	0.004	0.028	0.124	**0.770**	0.073
	6	0.000	0.000	0.000	0.003	0.074	**0.923**

P-screen Method

		Assigned Rank					
		1	2	3	4	5	6
True Rank	1	**0.613**	0.260	0.099	0.027	0.001	0.000
	2	0.258	**0.406**	0.245	0.082	0.008	0.001
	3	0.105	0.247	**0.394**	0.224	0.031	0.000
	4	0.024	0.084	0.237	**0.554**	0.099	0.002
	5	0.001	0.003	0.025	0.110	**0.791**	0.070
	6	0.000	0.000	0.001	0.003	0.069	**0.928**

Table 6

Probability of Correctly Ranking Each Task:
Type 2 Distribution with J=100 and GSD=4.

Multiple Linear Regression

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.572**	0.267	0.128	0.030	0.003	0.000
	2	0.259	**0.383**	0.264	0.090	0.004	0.000
True	3	0.146	0.269	**0.372**	0.195	0.018	0.000
Rank	4	0.022	0.081	0.223	**0.618**	0.055	0.000
	5	0.000	0.000	0.013	0.066	**0.902**	0.019
	6	0.000	0.000	0.000	0.000	0.019	**0.981**

P-screen Method

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.575**	0.264	0.127	0.032	0.002	0.000
	2	0.269	**0.390**	0.253	0.086	0.002	0.000
True	3	0.129	0.267	**0.381**	0.208	0.014	0.000
Rank	4	0.026	0.078	0.232	**0.618**	0.047	0.000
	5	0.001	0.001	0.008	0.055	**0.917**	0.018
	6	0.000	0.000	0.000	0.000	0.018	**0.982**

Table 7

Probability of Correctly Ranking Each Task:
Type 1 Distribution with J=100 and GSD=2.

Multiple Linear Regression

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.728**	0.242	0.028	0.002	0.000	0.000
	2	0.241	**0.607**	0.129	0.023	0.000	0.000
True	3	0.030	0.134	**0.667**	0.167	0.001	0.000
Rank	4	0.001	0.016	0.175	**0.773**	0.034	0.0000
	5	0.0000	0.0000	0.0003	0.035	**0.957**	0.007
	6	0.0000	0.0000	0.0000	0.000	0.007	**0.993**

P-screen Method

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.680**	0.275	0.041	0.003	0.000	0.000
	2	0.286	**0.579**	0.111	0.024	0.000	0.000
True	3	0.032	0.132	**0.661**	0.174	0.001	0.000
Rank	4	0.001	0.013	0.186	**0.774**	0.025	0.000
	5	0.000	0.000	0.001	0.025	**0.968**	0.006
	6	0.000	0.000	0.000	0.000	0.006	**0.994**

Table 8

Probability of Correctly Ranking Each Task:
Type 2 Distribution with J=100 and GSD=2.

Multiple Linear Regression

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.653**	0.267	0.072	0.007	0.000	0.000
	2	0.280	**0.523**	0.162	0.035	0.000	0.000
True	3	0.063	0.189	**0.582**	0.166	0.000	0.000
Rank	4	0.004	0.020	0.184	**0.779**	0.013	0.000
	5	0.000	0.000	0.000	0.013	**0.987**	0.000
	6	0.000	0.000	0.000	0.000	0.000	**1.000**

P-screen Method

		Assigned Rank					
		1	2	3	4	5	6
	1	**0.580**	0.308	0.096	0.017	0.000	0.000
	2	0.343	**0.477**	0.143	0.037	0.000	0.000
True	3	0.074	0.183	**0.571**	0.172	0.000	0.000
Rank	4	0.004	0.032	0.191	**0.768**	0.005	0.000
	5	0.000	0.000	0.000	0.005	**0.994**	0.000
	6	0.000	0.000	0.000	0.000	0.000	**0.999**

indicated that at least one MI data set was not equal to any others in each experiment. Therefore, there were statistically significant differences in performance between task time matrices (using MIs).

Correlation coefficients of the MIs (using 3000 computed values) for the two methods were also calculated for each experiment (Table 12). The MI correlation coefficients ranged from 0.66 - 0.86. Correlation increased with increasing number of samples, overlap, and GSD. The magnitude of the correlation coefficients indicated that although there was a tendency for both methods to misclassify the same runs, the degree of misclassification was not the same for both methods.

A review of the estimated task medians showed that the P-screen method was positively biased on the higher concentration tasks and the multiple linear regression method was also positively biased but for the lower concentration tasks. The estimated task medians for the regression method were "compressed" and the P-screen estimates were "expanded."

Table 9

Number of runs (out of 3000) that the highest two tasks were correctly ranked.

Type 1 Distribution

GSD=2

# Samples	Regression	P-Screen
100	2872	2905
20	1469	1517

GSD=4

# Samples	Regression	P-Screen
100	2308	2371
20	843	743

Type 2 Distribution

GSD=2

# Samples	Regression	P-Screen
100	2960	2983
20	1862	1830

GSD=4

# Samples	Regression	P-Screen
100	2705	2750
20	1171	1043

Table 10

Number of runs (out of 3000) that were correctly ranked.

Type 1 Distribution

GSD=2

# Samples	Regression	P-Screen
100	1377	1313
20	135	180

GSD=4

# Samples	Regression	P-Screen
100	542	547
20	79	66

Type 2 Distribution

GSD=2

# Samples	Regression	P-Screen
100	1173	1031
20	162	177

GSD=4

# Samples	Regression	P-Screen
100	601	618
20	100	91

Table 11

Misclassification Indicies*

Type 1 Distribution

GSD=2

# Samples	Regression	P-Screen
100	0.0766	0.0813
20	0.2880	0.2745

GSD=4

# Samples	Regression	P-Screen
100	0.1646	0.1599
20	0.3694	0.3794

Type 2 Distribution

GSD=2

# Samples	Regression	P-Screen
100	0.0938	0.1050
20	0.2774	0.2663

GSD=4

# Samples	Regression	P-Screen
100	0.1538	0.1506
20	0.3361	0.3452

*Average misclassification indices for 3000 runs.

Table 12

Correlation Coefficients*

Type 1 Distribution

GSD=2

# Samples	Correlation
100	0.749
20	0.724

GSD=4

# Samples	Correlation
100	0.863
20	0.784

Type 2 Distribution

GSD=2

# Samples	Correlation
100	0.706
20	0.661

GSD=4

# Samples	Correlation
100	0.842
20	0.756

*Correlation coefficients of the two methods using 3000 misclassifications indices.

CHAPTER V

DISCUSSION AND CONCLUSION

Multiple linear regression, used in many professions for many years, has been accepted as the preferred data analysis method to establish relationships between independent and dependent variables. The observation (based on the summary statistics) that the P-screen method performed similarly may be of interest to anyone conducting data analysis because each method has assumptions and limitations. Assumptions of multiple regression include: the model is linear, and the error terms are independent, have constant variance, and are normally distributed.[20] The P-screen method will not work for those jobs that require each task to be performed to complete the job (assembly line). For each sample, at least one task cannot be performed so that the number of unknowns (task medians) equals the number of simultaneous equations to be solved. Those samples for which all tasks were performed would have to be removed from the data set prior to analysis.

An understanding of each of the methods' assumptions and limitations allows someone to add to his data analysis "tool box." Furthermore, the comparative similar performance may indicate that a "revised P-screen" or some other alternative method may exist which can outperform the multiple linear regression or at least may not be subject to the same assumptions and limitations.

While the estimated versus actual ranking probabilities (Tables 1 - 8) tracked very closely, it would be necessary to conduct several replicates of the experiment to determine whether the small differences in performance (summary statistics) are statistically significant. The correlation coefficients of the MIs suggested that one method may perform better under certain conditions; however, the analyses conducted during this study did not identify these factors.

All statistical data analysis methods have limitations. It is important for the occupational hygienist to understand these limitations. The ability to check the results using two methods (as compared to only checking model assumptions of one method) could provide a more complete understanding of the data. In particular, this could be the case if the specific conditions under which one method performs better were known.

Important occupational health decisions are made based on the results of data analysis such as: whether or not to install engineering controls, whether or not personal protective equipment should be required, and whether the operation is in regulatory agency compliance. Any opportunities to improve data analysis techniques should be explored since the improvements may further protect the worker's health. As an occupational health program strategy moves away from being compliance driven towards a comprehensive exposure assessment program, data analysis will become a more integral part of the hygienist's duties. Use of a task ranking method could help advance an occupational health program in two ways: (1) it could identify high exposure tasks from historical multi-task data, and (2) it could reduce the total number of samples required to comprehensively assess all potential hazards. Any improvements in data analysis techniques could enhance the decision making of these programs.

BIBLIOGRAPHY

1. National Safety Council (NSC): *Fundamentals of Industrial Hygiene.* 2nd Edition. Edited by Julian B. Olishifski. Chicago, Ill: NSC, 1979. p. 2.

2. Burstyn, I. and Teschke, K. Studying the Determinants of Exposure: A Review of Methods. *Am. Ind. Hyg. Assoc. J. 60:* 57-72 (1999).

3. Garrison, R.P. and Erig, M. Ventilation to Eliminate Oxygen Deficiency in a Confined Space - Part III: Heavier-than-air characteristics. *Appl. Occup. Environ. Hyg. 6:* 131-140 (1991).

4. Heinonen, K., Kulmala, I., and Saamanen, A. Local Ventilation for Power Handling - Combination of Local Supply and Exhaust Air. *Am. Ind. Hyg. Assoc. J. 57:* 356-364 (1996).

5. Methner, M..M. and Fenske, R.A. Pesticide Exposure during Greenhouse Applications. III. Variable Exposure due to Ventilation Conditions and Spray Pressure. *Appl. Occup. Environ. Hyg. 11:* 174-180 (1996).

6. Thorpe, A. and Brown, R.C. Measurement of the Effectiveness of Dust Extraction Systems of Hand Sanders used on Wood. *Ann. Occup. Hyg. 38:* 279-302 (1994).

7. Greenspan, C.A., Moure-Eraso, R. Wegman, D.H., et al. Occupational Hygiene Characterization of a Highway Construction Project: A Pilot Study. *Appl. Occup. Environ. Hyg. 10:* 50-58 (1995).

8. Hansen, D.J. and Whitehead, L.W. The Influence of Task and Location on Solvent Exposures in a Printing Plant. *Am. Ind. Hyg. Assoc. J. 49:* 259-265 (1988).

9. Nieuwenhuijsen, M.J., Sandiford, C., Lowson, D., et al. Peak Exposure Concentrations of Dust and Flour Aeroallergen in Flour Mills and Bakeries. *Ann. Occ. Hyg. 39*: 193-201 (1995).

10. Smid, T., Heederik, D., Mensink, G., et al. Exposure to Dust, Endotoxin, and Fungi in the Animal Feed Industry. *Am. Ind. Hyg. Assoc. J. 53:* 362-368 (1992).

11. McDevitt, J.J., Lees, P.S.J., and McDiarmid, M.A. Exposure of Hospital Pharmacists and Nurses to Antineoplastic Agents. *J. Occup. Med. 35:* 57-60 (1990).

12. Scheeper, B., Kromhout, H. and Boleij, J.S.M. Wood-Dust Exposure during Wood-Working Processes. *Ann. Occ. Hyg. 39*: 141-154 (1995).

13. Williams, T.M., Levine, R.J. and Blunden, P.B. Exposure of Embalmers to Formaldehyde and other Chemicals. *Am. Ind. Hyg. Assoc. J. 45:* 172-176 (1984).

14. American Industrial Hygiene Association (AIHA): *The Occupational Environment-Its Evaluation and Control.* Edited by Salvatore R. DiNardi. Fairfax, VA:AIHA, 1998. p. 305.

15. American Conference of Governmental Industrial Hygienists (ACGIH). *Threshold Limit Values for Chemical Substances and Physical Agents, Biological Exposure Indices*. Cinncinnati, OH: ACGIH, 1998.

16. Burstyn, I., Teschke, K. and Kennedy, S.M. Exposure Levels and Determinants of Inhalable Dust Exposure Levels in Bakeries. *Ann. Occ. Hyg. 41*: 609-624 (1997).

17. Preller, L., Kromhout, H., Heederik, D. and Tielen, M. J. M. Modeling Long-Term Exposure in Occupational Exposure-Response Analysis. *Scand. J. Work Environ. Health. 21*: 504-12 (1995).

18. Kromhout, H., Swuste, P. and Boleij, J. S. M. Empirical Modelling of Chemical Exposure in the Rubber-Manufacturing Industry. *Ann. Occ. Hyg. 38*: 3-22 (1994).

19. Teschke, K., Marion, S.A., van Zuylen, M.J.A., et al. Maintenance of Stellite and Tungsten Carbide Saw Tips: Determinants of Exposure to Cobalt and Chromium. *Am. Ind. Hyg. Assoc. J. 56:* 661-668 (1995).

20. Wiley Series in Probability and Statistics: Applied Probability and Statistics Section. *Biostatistics: A Foundation for Analysis in the Health Sciences.* 7th Edition. Wayne M. Daniel. John Wiley & Sons, Inc., 1999. p. 475.

21. Phillips, M.L. and Esmen, N.A. Computational Method for Ranking Task-Specific Exposures using Multi-Task Time-Weighted Average Samples. *Ann. Occ. Hyg. 43:* 201-213 (1999).

22. Esmen, N.A. Classification of Worker Exposures. Promoting a Healthy Work Environment, Occupational Hygiene 1999. The British Occupational Hygiene Society.

APPENDIX A

ALGORITHMS OF MULTIPLE LINEAR REGRESSION MATRIX AND VECTORS WITH AN EXAMPLE OF A SOLUTION

$$\overleftrightarrow{LRA} = \begin{array}{ccccccc} m & \sum x_1 & \sum x_2 & \sum x_3 & \sum x_4 & \sum x_5 & \sum x_6 \\ \sum x_1 & \sum x_1^2 & \sum x_1x_2 & \sum x_1x_3 & \sum x_1x_4 & \sum x_1x_5 & \sum x_1x_6 \\ \sum x_2 & \sum x_2x_1 & \sum x_2^2 & \sum x_2x_3 & \sum x_2x_4 & \sum x_2x_5 & \sum x_2x_6 \\ \sum x_3 & \sum x_3x_1 & \sum x_3x_2 & \sum x_3^2 & \sum x_3x_4 & \sum x_3x_5 & \sum x_3x_6 \\ \sum x_4 & \sum x_4x_1 & \sum x_4x_2 & \sum x_4x_3 & \sum x_4^2 & \sum x_4x_5 & \sum x_4x_6 \\ \sum x_5 & \sum x_5x_1 & \sum x_5x_2 & \sum x_5x_3 & \sum x_5x_4 & \sum x_5^2 & \sum x_5x_6 \\ \sum x_6 & \sum x_6x_1 & \sum x_6x_2 & \sum x_6x_3 & \sum x_6x_4 & \sum x_6x_5 & \sum x_6^2 \end{array}$$

$$\overrightarrow{LRB} = \begin{array}{c} \sum y \\ \sum x_1y \\ \sum x_2y \\ \sum x_3y \\ \sum x_4y \\ \sum x_5y \\ \sum x_6y \end{array}$$

$$\overrightarrow{X} = \begin{array}{c} \alpha \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \\ \beta_6 \end{array}$$

EXAMPLE OF A MULTIPLE LINEAR REGRESSION SOLUTION.

THE SOLUTION SHOWS THE STEPS REQUIRED (AS DESCRIBED IN METHODS AND MATERIALS) TO OBTAIN AN INTERCEPT AND REGRESSION COEFFICIENTS FOR A SCENARIO WITH TEN SAMPLES AND SIX TASKS.

$\overleftrightarrow{TASKOCCUR}$

	1	1	1	1	0	1
	0	1	1	1	1	1
	0	0	1	0	1	1
	1	0	1	1	0	1
	0	1	1	1	0	1
	1	0	1	1	0	1
	0	1	0	1	0	0
	0	1	1	1	0	1
	0	1	0	1	0	1
	1	1	1	1	0	1
Sum of X	4	7	8	9	2	9
Sum of X^2	4	7	8	9	2	9

$\longrightarrow$

$\overleftrightarrow{LRA}$

10	4	7	8	9	2	9
4	4	2	4	4	0	4
7	2	7	5	7	1	6
8	4	5	8	7	2	8
9	4	7	7	9	1	8
2	0	1	2	1	2	2
9	4	6	8	8	2	9

$\longrightarrow$

$\overleftrightarrow{LRA}^{-1}$

3	-6E-17	-4E-16	-3E-16	-2	-1	-1
-6E-16	1	0.5	-0.5	-0.5	0.5	3E-16
-2E-15	0.5	1	-5E-17	-1	6.2E-16	5E-16
-1E-15	-0.5	-1E-16	1.5	1E-15	-0.5	-1
-2	-0.5	-1	1E-16	3	1	5E-17
-1	0.5	2.6E-16	-0.5	1	1.5	3E-16
-1	0	0	-1	0	0	2

Sample concentrations
2.495781
3.384271
4.246408
3.33527
1.024439
1.03136
2.905559
4.322038
2.531066
2.78459

$\longrightarrow$

$\overrightarrow{LRB}$
28.1
9.65
19.4
22.6
23.8
7.63
25.2

Solution

$\overrightarrow{X}$:

	Rank	Actual
3.7677		
-0.033	3	1
0.4569	5	2
0.1422	4	3
-1.319	1	4
0.711	6	5
-0.374	2	6

APPENDIX B

COMPUTER PROGRAM USED TO COMPARE THE TWO METHODS

```
REM         LRST1005.BAS
REM This program uses more finely quantized task duration inputs (1/96 of shift) and
censors tasks with duration < 5% of sample period

REM Task concentration distribution is log-normal but dependent on averaging time.
REM The underlying short-time distributions are assumed to be second-order stationary
REM Successive short-time concentrations during a task are assumed to be totally
uncorrelated.
REM The nominal GM and GSD are for the task taking 1/6 of shift.
REM GSDs of task distributions may differ from each other within specified bounds.

REM In this version the GSDs of the 1/6 time weighted task concentration distributions
are allowed to differ from nominal value within some defined bounds

REM Samples are taken randomly from 5 different workers, each of whom has
characteristicly higher or lower than median exposure for each task
REM Uses estimated task thetas to calculate p-screened theta-hat matrix

REM The program allows multiple (i.set) task duration matrices to be used, with (cyc)
replications for each theta matrix.
REM The task duration inputs are generated using the "tasktime" subroutine

REM Assigment of exposures
    DEFSTR U
    DEFINT I-K
    DIM estvact(6, 6, 2)
    DIM workereffect(5)
    DIM weight(100, 6)
    DIM esam(100, 2)
    DIM th(6)
    DIM theta(100, 6)
    DIM m(6, 2)
    DIM time(100, 6)
    DIM ord(100, 6)
    DIM ran(100, 6)
    DIM pthat(6, 6)
    DIM cumrank(6, 2)
    DIM averank(6, 2)
    DIM worker(100)
    DIM p(100, 2)
```

```
     DIM pr(6, 2)
     DIM q(6, 2)
     DIM r(6, 2)
     DIM a(6, 7)
     DIM pscreen(6, 100)
     DIM zer(6)
     DIM GSD(2)
     DIM sigtask(6, 2)
     DIM sigdiff(6, 2)
     DIM rantask(6)
     DIM taskord(6)
     DIM arithmean(6, 2)
     DIM arithvar(6, 2)
     DIM TASKOCCUR(100, 6)
     DIM y(100), LRA(7, 7), LRB(7, 2)
     DIM LRAB(7, 8), LRx(7)
     DIM LRq(6, 2), LRr(6, 2)
     DIM LRcumrank(6, 2), LRaverank(6, 2)
     DIM LRqtemp(7, 2)
     DIM mlr(3000, 2), mps(3000, 2)
     DIM LRmedian(6, 2)

REM  $DYNAMIC
     DIM x(6)
REM  $DYNAMIC
     DIM h(6, 7)
REM  $DYNAMIC
     DIM xx(6)
REM  $DYNAMIC
     DIM i.to(6)
REM  $DYNAMIC
     DIM i.from(6)
REM  $DYNAMIC
     DIM b(6, 7)
REM  $DYNAMIC
     DIM e(6)
     i.fail = 0
     u.fail = ""
REM create range of task GSDs
     sig1 = 2: sig2 = 2
```

```
    sigdev = 0
    FOR i.t = 1 TO 6
    sigdiff(i.t, 1) = sig1 - sigdev * (1.4 - i.t * .4)
    sigdiff(i.t, 2) = sig2 - sigdev * (1.4 - i.t * .4)
    NEXT i.t
    OPEN "c:\Qb45\results\PSresult.txt" FOR OUTPUT AS #3
    OPEN "c:\Qb45\results\LRresult.txt" FOR OUTPUT AS #2
    OPEN "c:\Qb45\results\setmisav.txt" FOR OUTPUT AS #4
    OPEN "c:\Qb45\results\runsmis.txt" FOR OUTPUT AS #5
    OPEN "c:\Qb45\results\medians1.txt" FOR OUTPUT AS #6
    OPEN "c:\Qb45\results\medians2.txt" FOR OUTPUT AS #7
    WRITE #6, "TYPE - 1", " 1,  2,  4,  8,  16,  32 "
    WRITE #7, "TYPE - 2", "0.5,  1.4,  4,  11.3,  32,  90.5 "
    WRITE #4, "Dataset, Avg&var mlr1,Avg&var mlr2, Avg&var mps1,
Avg&varmps2"
    WRITE #5, "Run, mlr1, mlr2, mps1, mps2"

REM read work distributions:
    CLS
    sfac = 0
    FOR i = 1 TO 5
    workereffect(i) = sfac - LOG(1)
    sfac = sfac + (LOG(1)) / 2
    NEXT i
    runs = 0: dataset = 0: cumhi1 = 0: cumhi2 = 0
    LRcumhi1 = 0: LRcumhi2 = 0
    cummlr1 = 0: cummlr2 = 0: cummps1 = 0: cummps2 = 0


    FOR i.set = 1 TO 30
    dataset = dataset + 1
    WRITE #3,
    WRITE #3, "Dataset #", dataset
    WRITE #2,
    WRITE #2, "Dataset #", dataset

    ERASE esam
    ERASE th
    ERASE theta
    ERASE m
ERASE pthat
ERASE cumrank
ERASE averank
ERASE worker
ERASE p
ERASE pr
ERASE q
ERASE r
ERASE a
ERASE pscreen
ERASE zer
ERASE TASKOCCUR
ERASE LRAB
ERASE LRA
ERASE LRx
ERASE LRq
ERASE LRr
ERASE LRcumrank
ERASE LRaverank
ERASE LRqtemp
ERASE LRmedian
GOSUB tasktime:
GOSUB taskvar:

FOR i.sample = 1 TO 100
s = 0
FOR i.task = 1 TO 6
IF weight(i.sample, i.task) > 4.8 THEN
s = s + 1
TASKOCCUR(i.sample, i.task) = 1
ELSE
TASKOCCUR(i.sample, i.task) = 0
pscreen(i.task, i.sample) = 1
zer(i.task) = zer(i.task) + 1
END IF
NEXT i.task
thet = 1 / s
FOR i = 1 TO 6
IF weight(i.sample, i) > 4.8 THEN
theta(i.sample, i) = thet
```

```
        ELSE theta(i.sample, i) = 0
        END IF
        NEXT i
        NEXT i.sample

REM generate p-screened theta hats and mmedians - Medians are absolute.
REM The generated medians are used for comparison:
        WRITE #3, "P-screened theta hat matrix"
        FOR i.tr = 1 TO 6
        FOR i.tc = 1 TO 6
        FOR i.s = 1 TO 100
        pthat(i.tr, i.tc) = pthat(i.tr, i.tc) + theta(i.s, i.tc) * pscreen(i.tr, i.s)
        NEXT i.s
        pthat(i.tr, i.tc) = pthat(i.tr, i.tc) / zer(i.tr)
        NEXT i.tc
        WRITE #3, pthat(i.tr, 1), pthat(i.tr, 2), pthat(i.tr, 3), pthat(i.tr, 4), pthat(i.tr, 5),
pthat(i.tr, 6)
        m(i.tr, 1) = 2 ^ (i.tr - 1)
        NEXT i.tr
REM random assignment of GSD to task
        FOR i.ran = 1 TO 6
        i.ord = taskord(i.ran)
        sigtask(i.ord, 1) = sigdiff(i.ran, 1)
        sigtask(i.ord, 2) = sigdiff(i.ran, 2)
        NEXT i.ran
        FOR i.tr = 1 TO 6
        arithmean(i.tr, 1) = m(i.tr, 1) * EXP(LOG(sigtask(i.tr, 1)) * LOG(sigtask(i.tr, 1)) / 2)
        arithvar(i.tr, 1) = 16 * (arithmean(i.tr, 1) ^ 2) * (EXP((LOG(sigtask(i.tr, 1))) ^ 2) - 1)
        m(i.tr, 2) = 2 ^ (1.5 * i.tr - 2.5)
        arithmean(i.tr, 2) = m(i.tr, 2) * EXP(LOG(sigtask(i.tr, 2)) * LOG(sigtask(i.tr, 2)) / 2)
        arithvar(i.tr, 2) = 16 * (arithmean(i.tr, 2) ^ 2) * (EXP((LOG(sigtask(i.tr, 2))) ^ 2) - 1)
        NEXT i.tr

REM Setting up the LRA matrix
REM First Row

PRINT "The LRA Matrix"
LRA(1, 1) = 100
FOR J = 1 TO 100
ATEMP12 = TASKOCCUR(J, 1)
```

```
 LRA(1, 2) = ATEMP12 + LRA(1, 2)
ATEMP13 = TASKOCCUR(J, 2)
 LRA(1, 3) = ATEMP13 + LRA(1, 3)
ATEMP14 = TASKOCCUR(J, 3)
 LRA(1, 4) = ATEMP14 + LRA(1, 4)
ATEMP15 = TASKOCCUR(J, 4)
 LRA(1, 5) = ATEMP15 + LRA(1, 5)
ATEMP16 = TASKOCCUR(J, 5)
 LRA(1, 6) = ATEMP16 + LRA(1, 6)
ATEMP17 = TASKOCCUR(J, 6)
 LRA(1, 7) = ATEMP17 + LRA(1, 7)
NEXT J

LRA(2, 1) = LRA(1, 2)
LRA(3, 1) = LRA(1, 3)
LRA(4, 1) = LRA(1, 4)
LRA(5, 1) = LRA(1, 5)
LRA(6, 1) = LRA(1, 6)
LRA(7, 1) = LRA(1, 7)

REM  'LRA' matrix diagonal elements, except (1,1)

FOR J = 1 TO 100
ATEMP22 = TASKOCCUR(J, 1) ^ 2
 LRA(2, 2) = ATEMP22 + LRA(2, 2)
ATEMP33 = TASKOCCUR(J, 2) ^ 2
 LRA(3, 3) = ATEMP33 + LRA(3, 3)
ATEMP44 = TASKOCCUR(J, 3) ^ 2
 LRA(4, 4) = ATEMP44 + LRA(4, 4)
ATEMP55 = TASKOCCUR(J, 4) ^ 2
 LRA(5, 5) = ATEMP55 + LRA(5, 5)
ATEMP66 = TASKOCCUR(J, 5) ^ 2
 LRA(6, 6) = ATEMP66 + LRA(6, 6)
ATEMP77 = TASKOCCUR(J, 6) ^ 2
 LRA(7, 7) = ATEMP77 + LRA(7, 7)
NEXT J

REM Intermultiplied elements

FOR J = 1 TO 100
```

```
ATEMP23 = TASKOCCUR(J, 1) * TASKOCCUR(J, 2)
 LRA(2, 3) = ATEMP23 + LRA(2, 3)
ATEMP24 = TASKOCCUR(J, 1) * TASKOCCUR(J, 3)
 LRA(2, 4) = ATEMP24 + LRA(2, 4)
ATEMP25 = TASKOCCUR(J, 1) * TASKOCCUR(J, 4)
 LRA(2, 5) = ATEMP25 + LRA(2, 5)
ATEMP26 = TASKOCCUR(J, 1) * TASKOCCUR(J, 5)
 LRA(2, 6) = ATEMP26 + LRA(2, 6)
ATEMP27 = TASKOCCUR(J, 1) * TASKOCCUR(J, 6)
 LRA(2, 7) = ATEMP27 + LRA(2, 7)
ATEMP34 = TASKOCCUR(J, 2) * TASKOCCUR(J, 3)
 LRA(3, 4) = ATEMP34 + LRA(3, 4)
ATEMP35 = TASKOCCUR(J, 2) * TASKOCCUR(J, 4)
 LRA(3, 5) = ATEMP35 + LRA(3, 5)
ATEMP36 = TASKOCCUR(J, 2) * TASKOCCUR(J, 5)
 LRA(3, 6) = ATEMP36 + LRA(3, 6)
ATEMP37 = TASKOCCUR(J, 2) * TASKOCCUR(J, 6)
 LRA(3, 7) = ATEMP37 + LRA(3, 7)
ATEMP45 = TASKOCCUR(J, 3) * TASKOCCUR(J, 4)
 LRA(4, 5) = ATEMP45 + LRA(4, 5)
ATEMP46 = TASKOCCUR(J, 3) * TASKOCCUR(J, 5)
 LRA(4, 6) = ATEMP46 + LRA(4, 6)
ATEMP47 = TASKOCCUR(J, 3) * TASKOCCUR(J, 6)
 LRA(4, 7) = ATEMP47 + LRA(4, 7)
ATEMP56 = TASKOCCUR(J, 4) * TASKOCCUR(J, 5)
 LRA(5, 6) = ATEMP56 + LRA(5, 6)
ATEMP57 = TASKOCCUR(J, 4) * TASKOCCUR(J, 6)
 LRA(5, 7) = ATEMP57 + LRA(5, 7)
ATEMP67 = TASKOCCUR(J, 5) * TASKOCCUR(J, 6)
 LRA(6, 7) = ATEMP67 + LRA(6, 7)

NEXT J

LRA(3, 2) = LRA(2, 3)
LRA(4, 2) = LRA(2, 4)
LRA(4, 3) = LRA(3, 4)
LRA(5, 2) = LRA(2, 5)
LRA(5, 3) = LRA(3, 5)
LRA(5, 4) = LRA(4, 5)
LRA(6, 2) = LRA(2, 6)
LRA(6, 3) = LRA(3, 6)
LRA(6, 4) = LRA(4, 6)
LRA(6, 5) = LRA(5, 6)
LRA(7, 2) = LRA(2, 7)
LRA(7, 3) = LRA(3, 7)
LRA(7, 4) = LRA(4, 7)
LRA(7, 5) = LRA(5, 7)
LRA(7, 6) = LRA(6, 7)

REM SIMULATION WITH SOLUTIONS
REM Cycles

    a$(1) = "Type - 1": a$(2) = "Type - 2"
    LRct1 = 0: LRct2 = 0: LRhi1 = 0: LRhi2 = 0
    ct1 = 0: ct2 = 0: cyc = 0: hi1 = 0: hi2 = 0
    setmlr1 = 0: setmlr2 = 0: setmps1 = 0: setmps2 = 0

    FOR i.cyc = 1 TO 100

REM generate exposure distributions for type 1 and type 2
    RANDOMIZE (VAL(RIGHT$(TIME$, 2)))
    WRITE #2, "TASKOCCUR Matrix"
    WRITE #3, "Worker, Task Durations and Sample Concentrations"
    FOR i.s = 1 TO 100
    s1 = 0: s2 = 0: l2 = LOG(2)
    worker(i.s) = INT(5 * RND) + 1
    effect = workereffect(worker(i.s))
    FOR i.t = 1 TO 6
    p = RND: q = RND
    norm = SQR(-2 * LOG(p)) * COS(6.28318 * q)
    IF weight(i.s, i.t) > 0 THEN
    sg1 = LOG(EXP(SQR(LOG((arithvar(i.t, 1) / weight(i.s, i.t)) / (arithmean(i.t, 1) ^ 2)
+ 1))) - .5 + RND): sg2 = LOG(EXP(SQR(LOG((arithvar(i.t, 2) / weight(i.s, i.t)) /
(arithmean(i.t, 2) ^ 2) + 1))) - .5 + RND)
    median1 = LOG(arithmean(i.t, 1)) - (sg1 ^ 2) / 2: median2 = LOG(arithmean(i.t, 2))
- (sg2 ^ 2) / 2
    END IF
    s1 = s1 + weight(i.s, i.t) * EXP((sg1 * norm + effect + median1)) / 96
    s2 = s2 + weight(i.s, i.t) * EXP((sg2 * norm + effect + median2)) / 96
    NEXT i.t
```

```
      WRITE #3, worker(i.s), weight(i.s, 1), weight(i.s, 2), weight(i.s, 3), weight(i.s, 4),
weight(i.s, 5), weight(i.s, 6), s1, s2
      esam(i.s, 1) = LOG(s1)
      esam(i.s, 2) = LOG(s2)
      WRITE #2, TASKOCCUR(i.s, 1), TASKOCCUR(i.s, 2), TASKOCCUR(i.s, 3),
TASKOCCUR(i.s, 4), TASKOCCUR(i.s, 5), TASKOCCUR(i.s, 6), esam(i.s, 1),
esam(i.s, 2)

      NEXT i.s
      ERASE LRB

REM Generating the LRB vector
REM

PRINT "The LRB Vector"
FOR i = 1 TO 2
FOR J = 1 TO 100
B1TEMP = esam(J, i)
 LRB(1, i) = B1TEMP + LRB(1, i)
B2TEMP = esam(J, i) * TASKOCCUR(J, 1)
 LRB(2, i) = B2TEMP + LRB(2, i)
B3TEMP = esam(J, i) * TASKOCCUR(J, 2)
 LRB(3, i) = B3TEMP + LRB(3, i)
B4TEMP = esam(J, i) * TASKOCCUR(J, 3)
 LRB(4, i) = B4TEMP + LRB(4, i)
B5TEMP = esam(J, i) * TASKOCCUR(J, 4)
 LRB(5, i) = B5TEMP + LRB(5, i)
B6TEMP = esam(J, i) * TASKOCCUR(J, 5)
 LRB(6, i) = B6TEMP + LRB(6, i)
B7TEMP = esam(J, i) * TASKOCCUR(J, 6)
 LRB(7, i) = B7TEMP + LRB(7, i)
NEXT J
NEXT i

WRITE #2, "LRA Matrix", "LRB Matrix 1&2 "
FOR i = 1 TO 7
WRITE #2, LRA(i, 1), LRA(i, 2), LRA(i, 3), LRA(i, 4), LRA(i, 5), LRA(i, 6), LRA(i, 7),
LRB(i, 1), LRB(i, 2)
NEXT i
```

```
REM CREATING THE LRAB MATRIX TO SEND TO MATSOLV1

FOR i.type = 1 TO 2
 FOR i = 1 TO 7
  FOR ii = 1 TO 7
  LRAB(i, ii) = LRA(i, ii)
  NEXT ii
 NEXT i
FOR iii = 1 TO 7
 LRAB(iii, 8) = LRB(iii, i.type)
NEXT iii

GOSUB Matsolv1:

REM Assigning the solution vector LRx to LRq Matrix
FOR i.x = 1 TO 7
 LRqtemp(i.x, i.type) = LRx(i.x)
NEXT i.x
NEXT i.type

REM Calculating the task median concentrations LR method
FOR ii = 1 TO 2
FOR i = 2 TO 7

LRmedian(i - 1, ii) = EXP(LRqtemp(1, ii) + LRqtemp(i, ii))
NEXT i
NEXT ii

REM Changing the 7 element solution matrix to 6 - eliminating the intercepts

FOR ii = 1 TO 2
FOR i = 1 TO 6
 LRq(i, ii) = LRqtemp(i + 1, ii)
 NEXT i
NEXT ii



REM Ranking the Beta coefficients
```

```
        FOR ii = 1 TO 2
        FOR i = 1 TO 6
        s = 1
        FOR J = 1 TO 6
        IF LRq(i, ii) > LRq(J, ii) THEN
        s = s + 1
        END IF
        NEXT J
        LRr(i, ii) = s
        LRcumrank(i, ii) = LRcumrank(i, ii) + s
        NEXT i
        NEXT ii

        WRITE #2, a$(1), LRq(1, 1), LRq(2, 1), LRq(3, 1), LRq(4, 1), LRq(5, 1), LRq(6, 1)
        WRITE #2, a$(1), LRr(1, 1), LRr(2, 1), LRr(3, 1), LRr(4, 1), LRr(5, 1), LRr(6, 1)
        WRITE #2, a$(2), LRq(1, 2), LRq(2, 2), LRq(3, 2), LRq(4, 2), LRq(5, 2), LRq(6, 2)
        WRITE #2, a$(2), LRr(1, 2), LRr(2, 2), LRr(3, 2), LRr(4, 2), LRr(5, 2), LRr(6, 2)

REM Calculating the misclassification of each run for the LR method"

        FOR ii = 1 TO 2
        mlrsum = 0
        FOR i = 1 TO 6
        mlrtemp = ABS(i - LRr(i, ii))
        mlrsum = mlrsum + mlrtemp
        NEXT i
        mlr(runs + 1, ii) = mlrsum / 18
        NEXT ii

REM  Count correctly ranked cycles
        d2 = 0
        FOR i = 1 TO 6
        d2 = d2 + (LRr(i, 1) - i) * (LRr(i, 1) - i)
        NEXT i
        IF d2 = 0 THEN
        LRct1 = LRct1 + 1
        END IF
        IF LRr(6, 1) = 6 AND LRr(5, 1) = 5 THEN
        LRhi1 = LRhi1 + 1
        END IF
        d2 = 0
        FOR i = 1 TO 6
        d2 = d2 + (LRr(i, 2) - i) * (LRr(i, 2) - i)
        NEXT i
        IF d2 = 0 THEN
        LRct2 = LRct2 + 1
        END IF
        IF LRr(6, 2) = 6 AND LRr(5, 2) = 5 THEN
        LRhi2 = LRhi2 + 1
        END IF
        FOR i.type = 1 TO 2
        FOR i.act = 1 TO 6
        FOR i.est = 1 TO 6
        IF LRr(i.act, i.type) = i.est THEN
        LRestvact(i.act, i.est, i.type) = LRestvact(i.act, i.est, i.type) + 1
        END IF
        NEXT i.est
        NEXT i.act
        NEXT i.type

REM find the overall variances:
        s1 = 0: s2 = 0: s12 = 0: s22 = 0
        FOR i = 1 TO 100
        s1 = s1 + esam(i, 1)
        s2 = s2 + esam(i, 2)
        s12 = s12 + esam(i, 1) * esam(i, 1)
        s22 = s22 + esam(i, 2) * esam(i, 2)
        NEXT i
        V(1) = (s12 - s1 * s1 / 100) / 99
        V(2) = (s22 - s2 * s2 / 100) / 99

REM Calculate Scaled values
        FOR i = 1 TO 2
        FOR ii = 1 TO 100
        p(ii, i) = esam(ii, i) / SQR(V(i))
        NEXT ii
        NEXT i
REM Calculate Sums
        WRITE #3, "P sums"
        FOR i = 1 TO 6
```

```
      s1 = 0: s2 = 0: s3 = 0
      FOR ii = 1 TO 100
REM test weight(i.sample, i.task) sum if it is less than 5% of sample period if not skip
      IF weight(ii, i) < 4.8 THEN
      s1 = s1 + p(ii, 1)
      s2 = s2 + p(ii, 2)
      s3 = s3 + 1
      END IF
      NEXT ii
      pr(i, 1) = s1 / s3
      pr(i, 2) = s2 / s3
      WRITE #3, pr(i, 1), pr(i, 2)
      NEXT i
REM compute results
      FOR i.spread = 1 TO 2
      FOR i.t = 1 TO 6
      FOR i.c = 1 TO 6
      a(i.t, i.c) = pthat(i.t, i.c)
      NEXT i.c
      a(i.t, 7) = pr(i.t, i.spread)
      PRINT a(i.t, 1), a(i.t, 2), a(i.t, 3), a(i.t, 4), a(i.t, 5), a(i.t, 6), a(i.t, 7)
      NEXT i.t
      GOSUB Matsolv:
      FOR i.t = 1 TO 6
      q(i.t, i.spread) = x(i.t)
      NEXT i.t
      NEXT i.spread

      FOR i = 1 TO 6
      FOR ij = 1 TO 2
      q(i, ij) = EXP(q(i, ij) * SQR(V(ij)))
      PRINT "   Type - ";
      PRINT USING "##"; ij;
      PRINT ":";
      PRINT USING "#####.###"; m(i, ij); q(i, ij);
      NEXT ij
      PRINT
      NEXT i
REM  ranking the calculated task concentrations
      FOR ii = 1 TO 2
```

```
        FOR i = 1 TO 6
        s = 1
        FOR J = 1 TO 6
        IF q(i, ii) > q(J, ii) THEN
        s = s + 1
        END IF
        NEXT J
        r(i, ii) = s
        cumrank(i, ii) = cumrank(i, ii) + s
        NEXT i
        NEXT ii

        WRITE #3, a$(1), q(1, 1), q(2, 1), q(3, 1), q(4, 1), q(5, 1), q(6, 1)
        WRITE #3, a$(1), r(1, 1), r(2, 1), r(3, 1), r(4, 1), r(5, 1), r(6, 1)
        WRITE #3, a$(2), q(1, 2), q(2, 2), q(3, 2), q(4, 2), q(5, 2), q(6, 2)
        WRITE #3, a$(2), r(1, 2), r(2, 2), r(3, 2), r(4, 2), r(5, 2), r(6, 2)

        WRITE #6, "PScreen", q(1, 1), q(2, 1), q(3, 1), q(4, 1), q(5, 1), q(6, 1)
        WRITE #6, "Lin reg", LRmedian(1, 1), LRmedian(2, 1), LRmedian(3, 1),
LRmedian(4, 1), LRmedian(5, 1), LRmedian(6, 1)

        WRITE #7, "PScreen", q(1, 2), q(2, 2), q(3, 2), q(4, 2), q(5, 2), q(6, 2)
        WRITE #7, "Lin reg", LRmedian(1, 2), LRmedian(2, 2), LRmedian(3, 2),
LRmedian(4, 2), LRmedian(5, 2), LRmedian(6, 2)


REM Calculating the misclassification of the PS method

        FOR ii = 1 TO 2
        mpssum = 0
        FOR i = 1 TO 6
        mpstemp = ABS(i - r(i, ii))
        mpssum = mpssum + mpstemp
        NEXT i
        mps(runs + 1, ii) = mpssum / 18
        NEXT ii
        WRITE #5, runs + 1, mlr(runs + 1, 1), mlr(runs + 1, 2), mps(runs + 1, 1), mps(runs +
1, 2)

REM  Count correctly ranked cycles
```

```
d2 = 0
FOR i = 1 TO 6
d2 = d2 + (r(i, 1) - i) * (r(i, 1) - i)
NEXT i
IF d2 = 0 THEN
ct1 = ct1 + 1
END IF
IF r(6, 1) = 6 AND r(5, 1) = 5 THEN
hi1 = hi1 + 1
END IF
d2 = 0
FOR i = 1 TO 6
d2 = d2 + (r(i, 2) - i) * (r(i, 2) - i)
NEXT i
IF d2 = 0 THEN
ct2 = ct2 + 1
END IF
IF r(6, 2) = 6 AND r(5, 2) = 5 THEN
hi2 = hi2 + 1
END IF
FOR i.type = 1 TO 2
FOR i.act = 1 TO 6
FOR i.est = 1 TO 6
IF r(i.act, i.type) = i.est THEN
estvact(i.act, i.est, i.type) = estvact(i.act, i.est, i.type) + 1
END IF
NEXT i.est
NEXT i.act
NEXT i.type

setmlr1 = setmlr1 + mlr(runs + 1, 1)
setmlr2 = setmlr2 + mlr(runs + 1, 2)
setmps1 = setmps1 + mps(runs + 1, 1)
setmps2 = setmps2 + mps(runs + 1, 2)
cyc = cyc + 1
runs = runs + 1
NEXT i.cyc
FOR ii = 1 TO 6
FOR i = 1 TO 2
averank(ii, i) = cumrank(ii, i) / cyc
```

```
    NEXT i
    WRITE #3, "task", ii, "average rank - type 1", averank(ii, 1), "average rank - type 2",
averank(ii, 2)
    PRINT "task"; ii; "average rank - type 1"; averank(ii, 1); "average rank - type 2";
averank(ii, 2)
    NEXT ii
    WRITE #3, ct1, "of", cyc, "Type 1 cycles correct"
    WRITE #3, ct2, "of", cyc, "Type 2 cycles correct"
    WRITE #3, hi1, "of", cyc, "highest 2 conc. correctly ranked for Type 1"
    WRITE #3, hi2, "of", cyc, "highest 2 conc. correctly ranked for Type 2"
    cumhi1 = cumhi1 + hi1
    cumhi2 = cumhi2 + hi2
    cumct1 = cumct1 + ct1
    cumct2 = cumct2 + ct2
    cummps1 = cummps1 + setmps1
    cummps2 = cummps2 + setmps2

    IF i.fail = 0 THEN
    ELSE
    WRITE #3, u.fail, i.fail, "times"
    END IF
    PRINT ct1; "of"; cyc; "type 1 cycles correct"
    PRINT
    PRINT hi1; "of"; cyc; "highest 2 conc. correctly ranked for Type 1"
    PRINT
    PRINT ct2; "of"; cyc; "type 2 cycles correct"
    PRINT
    PRINT hi2; "of"; cyc; "highest 2 conc. correctly ranked for Type 2"
    PRINT
    avgmps1 = setmps1 / cyc
    avgmps2 = setmps2 / cyc

REM Calculating the variance of the 100 misclassifications, P screen, per dataset
    k = runs - 99
    l = runs
    eps1 = 0: eps2 = 0: vsumps1 = 0
    vsumps2 = 0: vps1 = 0: vps2 = 0
    FOR i = k TO l
    eps1 = (mps(i, 1) - avgmps1) ^ 2
    eps2 = (mps(i, 2) - avgmps2) ^ 2
```

```
	vsumps1 = vsumps1 + eps1
	vsumps2 = vsumps2 + eps2
	NEXT i
	vps1 = vsumps1 / (cyc - 1)
	vps2 = vsumps2 / (cyc - 1)

	WRITE #3,
	FOR ii = 1 TO 6
	FOR i = 1 TO 2
	LRaverank(ii, i) = LRcumrank(ii, i) / cyc
	NEXT i
	WRITE #2, "task", ii, "average rank - type 1", LRaverank(ii, 1), "average rank - type
2", LRaverank(ii, 2)
	PRINT "task"; ii; "average rank - type 1"; LRaverank(ii, 1); "average rank - type 2";
LRaverank(ii, 2)
	NEXT ii
	WRITE #2, LRct1, "of", cyc, "Type 1 cycles correct"
	WRITE #2, LRct2, "of", cyc, "Type 2 cycles correct"
	WRITE #2, LRhi1, "of", cyc, "highest 2 conc. correctly ranked for Type 1"
	WRITE #2, LRhi2, "of", cyc, "highest 2 conc. correctly ranked for Type 2"
	LRcumhi1 = LRcumhi1 + LRhi1
	LRcumhi2 = LRcumhi2 + LRhi2
	LRcumct1 = LRcumct1 + LRct1
	LRcumct2 = LRcumct2 + LRct2
	cummlr1 = cummlr1 + setmlr1
	cummlr2 = cummlr2 + setmlr2

	IF i.fail = 0 THEN
	ELSE
	WRITE #2, u.fail, i.fail, "times"
	END IF
	PRINT LRct1; "of"; cyc; "type 1 cycles correct"
	PRINT
	PRINT LRhi1; "of"; cyc; "highest 2 conc. correctly ranked for Type 1"
	PRINT
	PRINT LRct2; "of"; cyc; "type 2 cycles correct"
	PRINT
	PRINT LRhi2; "of"; cyc; "highest 2 conc. correctly ranked for Type 2"
	PRINT
	avgmlr1 = setmlr1 / cyc
	avgmlr2 = setmlr2 / cyc
REM Calculating the variance of the 100 misclassifications LR, per dataset
	y = runs - 99
	z = runs
	emr1 = o: emr2 = 0: vsummr1 = 0
	vsummr2 = 0: vmr1 = 0: vmr2 = 0
	FOR i = y TO z
	emr1 = (mlr(i, 1) - avgmlr1) ^ 2
	emr2 = (mlr(i, 2) - avgmlr2) ^ 2
	vsummr1 = vsummr1 + emr1
	vsummr2 = vsummr2 + emr2
	NEXT i
	vmr1 = vsummr1 / (cyc - 1)
	vmr2 = vsummr2 / (cyc - 1)

	WRITE #4, dataset, avgmlr1, vmr1, avgmlr2, vmr2, avgmps1, vps1, avgmps2, vps2
	WRITE #2,

	NEXT i.set

	WRITE #3, "RESULTS OF SCREENED TIME MATRIX INVERSION
METHOD", LRA(1, 1), "SAMPLES"
	WRITE #3,
	WRITE #3, "Assumes zero autocorrelation"
	WRITE #3, "GSD of underlying task distribution for task time weight 1/6 differed
from nominal value by +/-", sigdev
	WRITE #3,
	WRITE #3, "Estimated versus actual rank  for", dataset, "datasets with", cyc, "cycles
per dataset"
	WRITE #3, "Total of", runs, "runs"
	WRITE #3, "Worker effect: multiply median concentration by",
EXP(workereffect(1)), EXP(workereffect(2)), EXP(workereffect(3)),
EXP(workereffect(4)), EXP(workereffect(5))
	WRITE #3,
	GSD(1) = sig1: GSD(2) = sig2
	FOR iii = 1 TO 2
	WRITE #3, "Type", iii, "  -  nominal GSD = ", GSD(iii), "+/-", sigdev
	FOR i = 1 TO 6
	FOR ii = 1 TO 6
	estvact(i, ii, iii) = estvact(i, ii, iii) / runs
```

```
    NEXT ii
    WRITE #3, "true rank =", i, estvact(i, 1, iii), estvact(i, 2, iii), estvact(i, 3, iii),
estvact(i, 4, iii), estvact(i, 5, iii), estvact(i, 6, iii)
    NEXT i
    WRITE #3,
    NEXT iii
    WRITE #3, cumhi1, "of", runs, "highest 2 tasks correctly ranked for Type 1"
    WRITE #3, cumhi2, "of", runs, "highest 2 tasks correctly ranked for Type 2"
    WRITE #3, cumct1, "of", runs, "Type 1 runs ranked correctly"
    WRITE #3, cumct2, "of", runs, "Type 2 runs ranked correctly"
    totavgmps1 = cummps1 / runs
    totavgmps2 = cummps2 / runs
    WRITE #3, "Avg misclass for P-screen Type 1 -", totavgmps1, "Type 2 -",
totavgmps2

    CLOSE #3

    WRITE #2, "RESULTS OF MULTIPLE LINEAR REGRESSION METHOD",
LRA(1, 1), "SAMPLES"
    WRITE #2,
    WRITE #2, "Assumes zero autocorrelation"
    WRITE #2, "GSD of underlying task distribution for task time weight 1/6 differed
from nominal value by +/-", sigdev
    WRITE #2,
    WRITE #2, "Estimated versus actual rank  for", dataset, "datasets with", cyc, "cycles
per dataset"
    WRITE #2, "Total of", runs, "runs"
    WRITE #2, "Worker effect: multiply median concentration by",
EXP(workereffect(1)), EXP(workereffect(2)), EXP(workereffect(3)),
EXP(workereffect(4)), EXP(workereffect(5))
    WRITE #2,
    GSD(1) = sig1: GSD(2) = sig2
    FOR iii = 1 TO 2
    WRITE #2, "Type", iii, "  -  nominal GSD = ", GSD(iii), "+/-", sigdev
    FOR i = 1 TO 6
    FOR ii = 1 TO 6
    LRestvact(i, ii, iii) = LRestvact(i, ii, iii) / runs
    NEXT ii
    WRITE #2, "true rank =", i, LRestvact(i, 1, iii), LRestvact(i, 2, iii), LRestvact(i, 3,
iii), LRestvact(i, 4, iii), LRestvact(i, 5, iii), LRestvact(i, 6, iii)
```

```
        REM Check here for 6 OR 7 SIZE ESTVACT
        NEXT i
        WRITE #2,
        NEXT iii
        WRITE #2, LRcumhi1, "of", runs, "highest 2 tasks correctly ranked for Type 1"
        WRITE #2, LRcumhi2, "of", runs, "highest 2 tasks correctly ranked for Type 2"
        WRITE #2, LRcumct1, "of", runs, "Type 1 runs ranked correctly"
        WRITE #2, LRcumct2, "of", runs, "Type 2 runs ranked correctly"
        totavgmlr1 = cummlr1 / runs
        totavgmlr2 = cummlr2 / runs
        WRITE #2, "Avg misclass for LR method Type 1 -", totavgmlr1, "Type 2 -",
totavgmlr2

        CLOSE #2

END

Matsolv:
REM Solution of simultaneous equations
REM THE MAIN SUBROUTINE

REM Definitions

        ERASE h
        REDIM h(6, 7)
        ERASE i.from
        REDIM i.from(6)
        ERASE i.to
        REDIM i.to(6)
        ERASE b
        REDIM b(6, 7)
        ERASE e
        REDIM e(6)
        ERASE x
        REDIM x(6)
        ERASE xx
        REDIM xx(6)
        kr = 6
         b.max = 0
         FOR i = 1 TO kr
        FOR J = 1 TO kr + 1
        b(i, J) = a(i, J)
        NEXT J
        IF ABS(b(i, kr + 1)) > b.max THEN
        b.max = ABS(b(i, kr + 1))
        END IF
       NEXT i
       GOSUB Analyse:
      CLS
RETURN
REM
REM the main subroutine ends here
REM
condition:

      LOCATE 10, 1
      COLOR 11
      PRINT
"+-*\=^#@...+-*\=^#@...^#@...+-*\=^#@...^#@...+-*\=^#@...^#@...+-*\=^#@..."
      COLOR 12
      LOCATE 15, 7
      PRINT "ESTIMATING THE CONDITION NUMBER OF THE COEFFICIENT
MATRIX "
      LOCATE 16, 22
      PRINT "THIS MAY TAKE A SHORT WHILE."
      LOCATE 21, 1
      PRINT
"+-*\=^#@...+-*\=^#@...^#@...+-*\=^#@...^#@...+-*\=^#@...^#@...+-*\=^#@..."
      r.min = 1E+20
      FOR i = 1 TO kr - 1
      FOR J = i + 1 TO kr
      r.sum = 0
      FOR k = 1 TO kr
      r.sum = r.sum + ABS(a(i, k) - a(J, k))
      NEXT k
      IF r.sum < r.min THEN
      r.min = r.sum
      END IF
      IF r.sum = 1E-10 THEN
      u.type = "Singular"
```

```
J = kr
i = kr - 1
END IF
NEXT J
NEXT i
IF u.type <> "Singular" THEN
c.min = 1E+20
FOR i = 1 TO kr - 1
FOR J = i + 1 TO kr
c.sum = 0
FOR k = 1 TO kr
c.sum = c.sum + ABS(a(k, i) - a(k, J))
NEXT k
IF c.sum < c.min THEN
c.min = c.sum
END IF
IF c.sum = 1E-10 THEN
u.type = "Singular"
J = kr
i = kr - 1
END IF
NEXT J
NEXT i
END IF
IF u.type <> "Singular" THEN
r.max = 0
FOR i = 1 TO kr
r.sum = 0
FOR k = 1 TO kr
r.sum = r.sum + ABS(a(i, k))
NEXT k
IF r.sum > r.max THEN
r.max = r.sum
END IF
NEXT i
END IF
IF u.type <> "Singular" THEN
c.max = 0
FOR i = 1 TO kr
c.sum = 0
```

```
        FOR k = 1 TO kr
        c.sum = c.sum + ABS(a(k, i))
        NEXT k
        IF c.sum > c.max THEN
        c.max = c.sum
        END IF
        NEXT i
        END IF
        IF u.type <> "Singular" THEN
        c1 = r.max / r.min
        c2 = c.max / c.min
        IF c1 > c2 THEN
        condition.a = c1
        ELSE
        condition.a = c2
        END IF
        END IF
RETURN:
Analyse:
REM the structured analysis program

        PRINT "b.max"; b.max
        GOSUB Error.crit:
        GOSUB Pivot:
        IF u.type <> "Singular" THEN
        GOSUB condition:
        END IF
        IF u.type <> "Singular" THEN
        GOSUB Crout:
        ELSE
        PRINT "Singular System"
        END IF

        IF u.type = "Singular" OR u.type = "Solution" THEN
        ELSE
        PRINT "LU Decomposition fails the error test - Trying Gauss-Seidel"
        GOSUB Gauss:
        END IF

        IF u.type = "Singular" OR u.type = "Solution" THEN
        ELSE
        PRINT "Gauss-Seidel iteration fails the error test - Trying LU iteration"
        GOSUB Iterate:
        END IF

        IF u.type = "Solution" OR u.type = "Best solution" THEN
        GOSUB Results:
        END IF

RETURN
Pivot:
REM This is pivoting and equilisation programme
        FOR in.dex = 1 TO kr
        IF in.dex < kr THEN
        a.max = 0
        FOR i = in.dex TO kr
        FOR J = in.dex TO kr
        IF ABS(a(i, J)) > a.max THEN
        a.max = ABS(a(i, J))
        i.m = i: j.m = J
        END IF
        NEXT J
        NEXT i
REM Exchange rows
        FOR i = 1 TO kr + 1
        SWAP a(in.dex, i), a(i.m, i)
        NEXT i
        i.to(in.dex) = in.dex
        i.from(in.dex) = j.m
        FOR J = 1 TO kr
        SWAP a(J, in.dex), a(J, j.m)
        NEXT J
        END IF
REM equilise
        a.max = 0
        FOR i = 1 TO kr
        IF ABS(a(in.dex, i)) > a.max THEN
        a.max = ABS(a(in.dex, i))
        END IF
        NEXT i
        IF a.max < 1E-22 THEN
        u.type = "Singular"
        PRINT "Failed as singular"
        PRINT "in.dex"
        in.dex = kr
        ELSE
        FOR i = 1 TO kr + 1
        a(in.dex, i) = a(in.dex, i) / a.max
        NEXT i
        END IF
        NEXT in.dex
RETURN
Error.crit:
REM error criteria
        e.rror = 1.45E-06
RETURN
Matrix:
        FOR i = 1 TO kr
        FOR J = 1 TO kr + 1
        IF i = J THEN
          ic = i - 1: pf = 1
        ELSEIF i < J THEN
          ic = i - 1: pf = h(i, i)
        ELSE
          ic = J - 1: pf = 1
        END IF
        pu = 0
        FOR k = 1 TO ic
        pu = pu + h(i, k) * h(k, J)
        NEXT k
        h(i, J) = (a(i, J) - pu) / pf
        IF i = J AND ABS(h(i, i)) < 1E-22 THEN
        PRINT " Singular "
        u.type = "Singular"
        i = kr
        J = kr + 1
        END IF
        NEXT J, i
RETURN
Crout:
```

```
REM LU decomposition - One pass
      u.method = "L/U Decomposition"
      GOSUB Matrix:
      IF u.type <> "Singular" THEN
      x(kr) = h(kr, kr + 1)
      FOR i = kr - 1 TO 1 STEP -1
      p0 = 0
      FOR J = i TO kr - 1
      p0 = p0 + h(i, J + 1) * x(J + 1)
      NEXT J
      x(i) = h(i, kr + 1) - p0
      NEXT i
      GOSUB res.check:
      average.error = e.sum / kr
      rms.error = SQR(e.ssq / kr)
      error.maximum = e.max
      IF e.max < e.rror THEN
      u.type = "Solution"
      ELSE
      GOSUB Failed:
      END IF
REM related to singular end if
      END IF

RETURN
Failed:
REM prints the failure
      CLS
      i.fail = i.fail + 1
      u.fail = "Results failed to meet the error criterion"
      WRITE #3, "Maximum residual =", error.maximum
RETURN
res.check:
REM       Check the results and print error values
REM swap back the solution vector
      e.max = 0
      FOR i = kr - 1 TO 1 STEP -1
      SWAP x(i.to(i)), x(i.from(i))
      NEXT i
REM check results with the original matrix
```

```
     e.sum = 0: e.ssq = 0
     FOR i = 1 TO kr
     e(i) = -b(i, kr + 1)
     FOR J = 1 TO kr
     e(i) = e(i) + b(i, J) * x(J)
     NEXT J
     IF ABS(e(i)) > e.max THEN
     e.max = ABS(e(i))
     END IF
     e.sum = e.sum + e(i)
     e.ssq = e.ssq + e(i) * e(i)
     NEXT i
     Rel.errmax = condition.a * e.max / b.max
     Rel.errmin = e.max / b.max / condition.a
RETURN
Swapback:
REM swap back
     FOR i = 1 TO kr - 1
     SWAP x(i.to(i)), x(i.from(i))
     NEXT i
RETURN
Gauss:
REM Gauss Seidel iteration
     u.method = "Gauss-Seidel Iteration"
     GOSUB Swapback
     e.init = e.max
     i.sei = 0
     FOR i = 1 TO kr
     xx(i) = x(i)
     NEXT i
     u.dur = "git"
     DO
     i.sei = i.sei + 1
     FOR i = 1 TO kr
     aso = a(i, kr + 1)
     FOR J = 1 TO kr
     IF i <> J THEN
     aso = aso - a(i, J) * x(J)
     END IF
     NEXT J
```

```
        x.new = aso / a(i, i)
        x(i) = x.new
        NEXT i
        GOSUB res.check:
        IF e.max < e.rror THEN
        u.dur = "dur"
        ELSEIF e.max > e.init THEN
        u.dur = "dur"
        GOSUB Swapback:
        FOR i = 1 TO kr
        x(i) = xx(i)
        NEXT i
        GOSUB res.check:
        ELSE
        e.init = e.max
        GOSUB Swapback:
        FOR i = 1 TO kr
        xx(i) = x(i)
        NEXT i
        u.dur = "git"
        END IF
        IF i.sei > 2 * kr THEN
        u.dur = "dur"
        END IF
        LOOP UNTIL u.dur = "dur"
        average.error = e.sum / kr
        rms.error = SQR(e.ssq / kr)
        error.maximum = e.max
        IF e.max < e.rror THEN
        u.type = "Solution"
        ELSE
        GOSUB Failed:
        END IF
RETURN
Iterate:
REM Lu decomposition with iteration
        u.method = "L/U Iteration"
        GOSUB recalc:
         u.lue = "go"
         e.old = e.max
```

```
FOR i = 1 TO kr
xx(i) = x(i)
NEXT i
DO
ed.max = 0
x.max = 0
GOSUB Matrix:
IF u.type <> "Singular" THEN
e(kr) = h(kr, kr + 1)
FOR i = kr - 1 TO 1 STEP -1
pi0 = 0
FOR J = i TO kr - 1
pi0 = pi0 + h(i, J + 1) * e(J + 1)
NEXT J
e(i) = h(i, kr + 1) - pi0
NEXT i
ELSE
FOR i = 1 TO kr
e(i) = .1
NEXT i
END IF
FOR i = 1 TO kr
x(i) = x(i) + e(i)
IF ABS(e(i)) > ed.max THEN
ed.max = ABS(e(i))
END IF
IF ABS(xx(i)) > x.max THEN
x.max = ABS(xx(i))
END IF
NEXT i
PRINT e.max
IF (ed.max / x.max) > .5 THEN
PRINT "The system is practically singular within the working accuracy"
PRINT "Failed as type 2 singular"
FOR i = 1 TO kr
x(i) = xx(i)
NEXT i
GOSUB res.check:
u.lue = "stop"
ELSE
```

```
          GOSUB res.check:
          IF e.max < e.rror THEN
          u.lue = "stop"
          ELSEIF e.max >= e.old THEN
          u.lue = "stop"
          FOR i = 1 TO kr
          x(i) = xx(i)
          NEXT i
          GOSUB res.check:
          ELSE
          GOSUB recalc:
          FOR i = 1 TO kr
          xx(i) = x(i)
          NEXT i
          u.lue = "go"
          END IF
          END IF
          LOOP UNTIL u.lue = "stop"

         average.error = e.sum / kr
         rms.error = SQR(e.ssq / kr)
         error.maximum = e.max
         IF e.max < e.rror THEN
         u.type = "Solution"
         ELSE
         u.type = "Best solution"
         END IF

RETURN
recalc:
REM swap back
         FOR i = 1 TO kr - 1
         SWAP x(i.to(i)), x(i.from(i))
         NEXT i
REM recalculates the constant vector for L/U iteration and overwrites
         FOR i = 1 TO kr
         FOR J = 1 TO kr
         a(i, kr + 1) = a(i, kr + 1) - a(i, J) * x(J)
         NEXT J
         NEXT i
```

```
RETURN
Results:
    CLS
    LOCATE 2, 5

    IF u.type = "Solution" THEN
    COLOR 10
    LOCATE 10, 10
    PRINT "*************** SOLUTION IS OBTAINED *****************"
    COLOR 15
    LOCATE 5, 2
    PRINT ""
    ELSE
    COLOR 12
    LOCATE 6, 10
    PRINT "THE BEST AVAILABLE SOLUTION WITHIN THE ERROR
SPECIFIED"
    COLOR 14
    PRINT "     The solution can only be improved by using ";
    PRINT "a solution program with "
    COLOR 13
    PRINT "           DOUBLE PRECISION ";
    COLOR 14
    PRINT "arithmetic "
    PRINT ""
    COLOR 12
    LOCATE 10, 10
    PRINT "*************** CURRENT SOLUTION  *****************"
    END IF
    GOSUB Show:

RETURN
Show:
REM prints results on the screen
    CLS
    LOCATE 5, 12
    PRINT "Method of Calculation : ";
    COLOR 11
    PRINT u.method
    WRITE #3, "calculated by", u.method
    COLOR 15
    LOCATE 7, 5
    PRINT "  RESULTS ( ";
    IF u.type = "Solution" THEN
    PRINT " Solution shown is within the specified error ) :"
    ELSE
    PRINT " Results shown failed to meet the error criterion ) :"
    END IF
    klef = kr MOD 3
    klim = kr - klef
    IF klim > 2 THEN
    FOR i = 1 TO klim
    PRINT " X("; i; ") = ";
    PRINT USING "##.######^^^^"; x(i);
    IF (i MOD 3) = 0 THEN
    PRINT ""
    END IF
    NEXT i
    ELSE
    klim = 0
    END IF
    IF klef = 0 THEN
    ELSE
    FOR i = klim + 1 TO kr
    PRINT " X("; i; ") = ";
    PRINT USING "##.######^^^^"; x(i);
    NEXT i
    END IF
    PRINT ""
    PRINT
"+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++"

    PRINT "Error analysis :"
    PRINT "Maximum residual Calculated : ";
    PRINT USING "##.######^^^^"; error.maximum
    PRINT "Average Residual Calculated : ";
    PRINT USING "##.######^^^^"; average.error
    PRINT "Root mean square Residual Calculated : ";
    PRINT USING "##.######^^^^"; rms.error
    PRINT "Coefficient Matrix Condition number estimate : ";
```

```
     PRINT USING "##.##^^^^"; condition.a
     PRINT ""
     PRINT USING "##.##^^^^"; Rel.errmin;
     PRINT " .LE. Relative Error .LE. ";
     PRINT USING "##.##^^^^"; Rel.errmax
     PRINT ""
     FOR i = 1 TO kr
     PRINT "Residual of equation "; i; " = ";
     PRINT USING "##.######^^^^"; e(i)
     NEXT i
     CLS
RETURN

Matsolv1:
REM Solution of simultaneous equations for Mult Linear Regression
REM THE MAIN SUBROUTINE

REM Definitions

     ERASE h
     REDIM h(7, 8)
     ERASE i.from
     REDIM i.from(7)
     ERASE i.to
     REDIM i.to(7)
     ERASE b
     REDIM b(7, 8)
     ERASE e
     REDIM e(7)
     ERASE LRx
REM     REDIM LRx(7)
     ERASE xx
     REDIM xx(7)
     kr = 7
      b.max = 0
      FOR i = 1 TO kr
       FOR J = 1 TO kr + 1
       b(i, J) = LRAB(i, J)
       NEXT J
       IF ABS(b(i, kr + 1)) > b.max THEN
```

```
          b.max = ABS(b(i, kr + 1))
          END IF
         NEXT i
         GOSUB LRAnalyse:
       CLS
RETURN
REM
REM the main subroutine ends here
REM
LRcondition:

       LOCATE 10, 1
       COLOR 11
       PRINT
"+-*\=^#@...+-*\=^#@...^#@...+-*\=^#@...^#@...+-*\=^#@...^#@...+-*\=^#@..."
       COLOR 12
       LOCATE 15, 7
       PRINT "ESTIMATING THE CONDITION NUMBER OF THE COEFFICIENT
MATRIX "
       LOCATE 16, 22
       PRINT "THIS MAY TAKE A SHORT WHILE."
       LOCATE 21, 1
       PRINT
"+-*\=^#@...+-*\=^#@...^#@...+-*\=^#@...^#@...+-*\=^#@...^#@...+-*\=^#@..."
       r.min = 1E+20
       FOR i = 1 TO kr - 1
       FOR J = i + 1 TO kr
       r.sum = 0
       FOR k = 1 TO kr
       r.sum = r.sum + ABS(LRAB(i, k) - LRAB(J, k))
       NEXT k
       IF r.sum < r.min THEN
       r.min = r.sum
       END IF
       IF r.sum = 1E-10 THEN
       u.type = "Singular"
       J = kr
       i = kr - 1
       END IF
       NEXT J
NEXT i
IF u.type <> "Singular" THEN
c.min = 1E+20
FOR i = 1 TO kr - 1
FOR J = i + 1 TO kr
c.sum = 0
FOR k = 1 TO kr
c.sum = c.sum + ABS(LRAB(k, i) - LRAB(k, J))
NEXT k
IF c.sum < c.min THEN
c.min = c.sum
END IF
IF c.sum = 1E-10 THEN
u.type = "Singular"
J = kr
i = kr - 1
END IF
NEXT J
NEXT i
END IF
IF u.type <> "Singular" THEN
r.max = 0
FOR i = 1 TO kr
r.sum = 0
FOR k = 1 TO kr
r.sum = r.sum + ABS(LRAB(i, k))
NEXT k
IF r.sum > r.max THEN
r.max = r.sum
END IF
NEXT i
END IF
IF u.type <> "Singular" THEN
c.max = 0
FOR i = 1 TO kr
c.sum = 0
FOR k = 1 TO kr
c.sum = c.sum + ABS(LRAB(k, i))
NEXT k
IF c.sum > c.max THEN
```

```
    c.max = c.sum
    END IF
    NEXT i
    END IF
    IF u.type <> "Singular" THEN
    c1 = r.max / r.min
    c2 = c.max / c.min
    IF c1 > c2 THEN
    condition.a = c1
    ELSE
    condition.a = c2
    END IF
    END IF
RETURN:
LRAnalyse:
REM the structured analysis program

    PRINT "b.max"; b.max
    GOSUB LRError.crit:
    GOSUB LRPivot:
    IF u.type <> "Singular" THEN
    GOSUB LRcondition:
    END IF
    IF u.type <> "Singular" THEN
    GOSUB LRCrout:
    ELSE
    PRINT "Singular System"
    END IF

    IF u.type = "Singular" OR u.type = "Solution" THEN
    ELSE
    PRINT "LU Decomposition fails the error test - Trying Gauss-Seidel"
    GOSUB LRGauss:
    END IF

    IF u.type = "Singular" OR u.type = "Solution" THEN
    ELSE
    PRINT "Gauss-Seidel iteration fails the error test - Trying LU iteration"
    GOSUB LRIterate:
    END IF
```

```
        IF u.type = "Solution" OR u.type = "Best solution" THEN
        GOSUB LRResults:
        END IF

RETURN
LRPivot:
REM This is pivoting and equilisation programme
        FOR in.dex = 1 TO kr
        IF in.dex < kr THEN
        a.max = 0
        FOR i = in.dex TO kr
        FOR J = in.dex TO kr
        IF ABS(LRAB(i, J)) > a.max THEN
        a.max = ABS(LRAB(i, J))
        i.m = i: j.m = J
        END IF
        NEXT J
        NEXT i
REM Exchange rows
        FOR i = 1 TO kr + 1
        SWAP LRAB(in.dex, i), LRAB(i.m, i)
        NEXT i
        i.to(in.dex) = in.dex
        i.from(in.dex) = j.m
        FOR J = 1 TO kr
        SWAP LRAB(J, in.dex), LRAB(J, j.m)
        NEXT J
        END IF
REM equilise
        a.max = 0
        FOR i = 1 TO kr
        IF ABS(LRAB(in.dex, i)) > a.max THEN
        a.max = ABS(LRAB(in.dex, i))
        END IF
        NEXT i
        IF a.max < 1E-22 THEN
        u.type = "Singular"
        PRINT "Failed as singular"
        PRINT "in.dex"
        in.dex = kr
        ELSE
        FOR i = 1 TO kr + 1
        LRAB(in.dex, i) = LRAB(in.dex, i) / a.max
        NEXT i
        END IF
        NEXT in.dex
RETURN
LRError.crit:
REM error criteria
        e.rror = 1.45E-06
RETURN
LRMatrix:
        FOR i = 1 TO kr
        FOR J = 1 TO kr + 1
        IF i = J THEN
          ic = i - 1: pf = 1
        ELSEIF i < J THEN
          ic = i - 1: pf = h(i, i)
        ELSE
          ic = J - 1: pf = 1
        END IF
        pu = 0
        FOR k = 1 TO ic
        pu = pu + h(i, k) * h(k, J)
        NEXT k
        h(i, J) = (LRAB(i, J) - pu) / pf
        IF i = J AND ABS(h(i, i)) < 1E-22 THEN
        PRINT " Singular "
        u.type = "Singular"
        i = kr
        J = kr + 1
        END IF
        NEXT J, i
RETURN
LRCrout:
REM LU decomposition - One pass
        u.method = "L/U Decomposition"
        GOSUB LRMatrix:
        IF u.type <> "Singular" THEN
        LRx(kr) = h(kr, kr + 1)
```

```
    FOR i = kr - 1 TO 1 STEP -1
    p0 = 0
    FOR J = i TO kr - 1
    p0 = p0 + h(i, J + 1) * LRx(J + 1)
    NEXT J
    LRx(i) = h(i, kr + 1) - p0
    NEXT i
    GOSUB LRres.check:
    average.error = e.sum / kr
    rms.error = SQR(e.ssq / kr)
    error.maximum = e.max
    IF e.max < e.rror THEN
    u.type = "Solution"
    ELSE
    GOSUB LRFailed:
    END IF
REM related to singular end if
    END IF

RETURN
LRFailed:
REM prints the failure
    CLS
    i.fail = i.fail + 1
    u.fail = "Results failed to meet the error criterion"
    WRITE #2, "Maximum residual =", error.maximum
RETURN
LRres.check:
REM     Check the results and print error values
REM swap back the solution vector
    e.max = 0
    FOR i = kr - 1 TO 1 STEP -1
    SWAP LRx(i.to(i)), LRx(i.from(i))
    NEXT i
REM check results with the original matrix
    e.sum = 0: e.ssq = 0
    FOR i = 1 TO kr
    e(i) = -b(i, kr + 1)
    FOR J = 1 TO kr
    e(i) = e(i) + b(i, J) * LRx(J)
```

```
     NEXT J
     IF ABS(e(i)) > e.max THEN
     e.max = ABS(e(i))
     END IF
     e.sum = e.sum + e(i)
     e.ssq = e.ssq + e(i) * e(i)
     NEXT i
     Rel.errmax = condition.a * e.max / b.max
     Rel.errmin = e.max / b.max / condition.a
RETURN
LRSwapback:
REM swap back
     FOR i = 1 TO kr - 1
     SWAP LRx(i.to(i)), LRx(i.from(i))
     NEXT i
RETURN
LRGauss:
REM Gauss Seidel iteration
     u.method = "Gauss-Seidel Iteration"
     GOSUB LRSwapback
     e.init = e.max
     i.sei = 0
     FOR i = 1 TO kr
     xx(i) = LRx(i)
     NEXT i
     u.dur = "git"
     DO
     i.sei = i.sei + 1
     FOR i = 1 TO kr
     aso = LRAB(i, kr + 1)
     FOR J = 1 TO kr
     IF i <> J THEN
     aso = aso - LRAB(i, J) * LRx(J)
     END IF
     NEXT J
     x.new = aso / LRAB(i, i)
     LRx(i) = x.new
     NEXT i
     GOSUB LRres.check:
     IF e.max < e.rror THEN
```

```
        u.dur = "dur"
        ELSEIF e.max > e.init THEN
        u.dur = "dur"
        GOSUB LRSwapback:
        FOR i = 1 TO kr
        LRx(i) = xx(i)
        NEXT i
        GOSUB LRres.check:
        ELSE
        e.init = e.max
        GOSUB LRSwapback:
        FOR i = 1 TO kr
        xx(i) = LRx(i)
        NEXT i
        u.dur = "git"
        END IF
        IF i.sei > 2 * kr THEN
        u.dur = "dur"
        END IF
        LOOP UNTIL u.dur = "dur"
        average.error = e.sum / kr
        rms.error = SQR(e.ssq / kr)
        error.maximum = e.max
        IF e.max < e.rror THEN
        u.type = "Solution"
        ELSE
        GOSUB LRFailed:
        END IF
RETURN
LRIterate:
REM Lu decomposition with iteration
        u.method = "L/U Iteration"
        GOSUB LRrecalc:
         u.lue = "go"
         e.old = e.max
         FOR i = 1 TO kr
         xx(i) = LRx(i)
         NEXT i
         DO
         ed.max = 0
```

```
x.max = 0
GOSUB LRMatrix:
IF u.type <> "Singular" THEN
e(kr) = h(kr, kr + 1)
FOR i = kr - 1 TO 1 STEP -1
pi0 = 0
FOR J = i TO kr - 1
pi0 = pi0 + h(i, J + 1) * e(J + 1)
NEXT J
e(i) = h(i, kr + 1) - pi0
NEXT i
ELSE
FOR i = 1 TO kr
e(i) = .1
NEXT i
END IF
FOR i = 1 TO kr
LRx(i) = LRx(i) + e(i)
IF ABS(e(i)) > ed.max THEN
ed.max = ABS(e(i))
END IF
IF ABS(xx(i)) > x.max THEN
x.max = ABS(xx(i))
END IF
NEXT i
PRINT e.max
IF (ed.max / x.max) > .5 THEN
PRINT "The system is practically singular within the working accuracy"
PRINT "Failed as type 2 singular"
FOR i = 1 TO kr
LRx(i) = xx(i)
NEXT i
GOSUB LRres.check:
u.lue = "stop"
ELSE
GOSUB LRres.check:
IF e.max < e.rror THEN
u.lue = "stop"
ELSEIF e.max >= e.old THEN
u.lue = "stop"
```

```
        FOR i = 1 TO kr
        LRx(i) = xx(i)
        NEXT i
        GOSUB LRres.check:
        ELSE
        GOSUB LRrecalc:
        FOR i = 1 TO kr
        xx(i) = LRx(i)
        NEXT i
        u.lue = "go"
        END IF
        END IF
        LOOP UNTIL u.lue = "stop"

       average.error = e.sum / kr
       rms.error = SQR(e.ssq / kr)
       error.maximum = e.max
       IF e.max < e.rror THEN
       u.type = "Solution"
       ELSE
       u.type = "Best solution"
       END IF

RETURN
LRrecalc:
REM swap back
       FOR i = 1 TO kr - 1
       SWAP LRx(i.to(i)), LRx(i.from(i))
       NEXT i
REM recalculates the constant vector for L/U iteration and overwrites
       FOR i = 1 TO kr
       FOR J = 1 TO kr
       LRAB(i, kr + 1) = LRAB(i, kr + 1) - LRAB(i, J) * LRx(J)
       NEXT J
       NEXT i
RETURN
LRResults:
       CLS
       LOCATE 2, 5
```

```
    IF u.type = "Solution" THEN
    COLOR 10
    LOCATE 10, 10
    PRINT "*************** SOLUTION IS OBTAINED *****************"
    COLOR 15
    LOCATE 5, 2
    PRINT ""
    ELSE
    COLOR 12
    LOCATE 6, 10
    PRINT "THE BEST AVAILABLE SOLUTION WITHIN THE ERROR
SPECIFIED"
    COLOR 14
    PRINT "     The solution can only be improved by using ";
    PRINT "a solution program with "
    COLOR 13
    PRINT "             DOUBLE PRECISION ";
    COLOR 14
    PRINT "arithmetic "
    PRINT ""
    COLOR 12
    LOCATE 10, 10
    PRINT "*************** CURRENT SOLUTION  *****************"
    END IF
    GOSUB LRShow:

RETURN
LRShow:
REM prints results on the screen
    CLS
    LOCATE 5, 12
    PRINT "Method of Calculation : ";
    COLOR 11
    PRINT u.method
    WRITE #2, "calculated by", u.method
    COLOR 15
    LOCATE 7, 5
    PRINT "  RESULTS ( ";
    IF u.type = "Solution" THEN
    PRINT " Solution shown is within the specified error ) :"
```

```
     ELSE
     PRINT " Results shown failed to meet the error criterion ) :"
     END IF
     klef = kr MOD 3
     klim = kr - klef
     IF klim > 2 THEN
     FOR i = 1 TO klim
     PRINT " X("; i; ") = ";
     PRINT USING "##.######^^^^"; LRx(i);
     IF (i MOD 3) = 0 THEN
     PRINT ""
     END IF
     NEXT i
     ELSE
     klim = 0
     END IF
     IF klef = 0 THEN
     ELSE
     FOR i = klim + 1 TO kr
     PRINT " X("; i; ") = ";
     PRINT USING "##.######^^^^"; LRx(i);
     NEXT i
     END IF
     PRINT ""
     PRINT
"++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++"

     PRINT "Error analysis :"
     PRINT "Maximum residual Calculated : ";
     PRINT USING "##.######^^^^"; error.maximum
     PRINT "Average Residual Calculated : ";
     PRINT USING "##.######^^^^"; average.error
     PRINT "Root mean square Residual Calculated : ";
     PRINT USING "##.######^^^^"; rms.error
     PRINT "Coefficient Matrix Condition number estimate : ";
     PRINT USING "##.##^^^^"; condition.a
     PRINT ""
     PRINT USING "##.##^^^^"; Rel.errmin;
     PRINT " .LE. Relative Error .LE. ";
     PRINT USING "##.##^^^^"; Rel.errmax
     PRINT ""
     FOR i = 1 TO kr
     PRINT "Residual of equation "; i; " = ";
     PRINT USING "##.######^^^^"; e(i)
     NEXT i
     CLS
RETURN


tasktime:
REM  Generate task durations

REM Generate task assignment matrix
100    ERASE ran
       ERASE weight
       ERASE ord
       ERASE time
       FOR ii = 1 TO 100
       RANDOMIZE (VAL(RIGHT$(TIME$, 2)))
       FOR i = 1 TO 6
       ran(ii, i) = RND
       NEXT i
       NEXT ii
       FOR ii = 1 TO 100
       FOR i = 1 TO 6
       s = 1
       FOR J = 1 TO 6
       IF ran(ii, i) > ran(ii, J) THEN
       s = s + 1
       END IF
       NEXT J
       ord(ii, i) = s
       NEXT i
       NEXT ii
REM Generate times matrix
       FOR ii = 1 TO 100
       J = 96
       FOR i = 1 TO 4
       jj = INT(J * RND)
       time(ii, i) = jj
```

```
        J = J - jj
        NEXT i
        time(ii, 5) = 96 - time(ii, 1) - time(ii, 2) - time(ii, 3) - time(ii, 4)
        NEXT ii
REM Assign times to tasks
        FOR ii = 1 TO 100
        FOR i = 1 TO 6
        J = ord(ii, i)
        weight(ii, J) = time(ii, i)
        NEXT i
        NEXT ii
REM  Check for samples with one task excluded
        s1 = 0: s2 = 0: s3 = 0: s4 = 0: s5 = 0: s6 = 0
        FOR ii = 1 TO 100
        ss = 0
        FOR i = 1 TO 6
        IF weight(ii, i) > 0 THEN
        ss = ss + 1
        END IF
        NEXT i
        IF ss = 5 THEN
        GOSUB task:

        END IF
        NEXT ii
        IF s1 * s2 * s3 * s4 * s5 * s6 = 0 THEN
        GOTO 100
        END IF

        RETURN


task:
        IF weight(ii, 1) = 0 THEN
        s1 = s1 + 1

        ELSEIF weight(ii, 2) = 0 THEN
        s2 = s2 + 1

        ELSEIF weight(ii, 3) = 0 THEN
        s3 = s3 + 1

        ELSEIF weight(ii, 4) = 0 THEN
        s4 = s4 + 1

        ELSEIF weight(ii, 5) = 0 THEN
        s5 = s5 + 1

        ELSEIF weight(ii, 6) = 0 THEN
        s6 = s6 + 1

        END IF
        RETURN

taskvar:
REM Creates random order for assigning GSD to task
        ERASE rantask
        ERASE taskord
        RANDOMIZE (VAL(RIGHT$(TIME$, 2)))
        FOR ii = 1 TO 6
        rantask(ii) = RND
        NEXT ii
        FOR ii = 1 TO 6
        s = 1
        FOR J = 1 TO 6
        IF rantask(ii) > rantask(J) THEN
        s = s + 1
        END IF
        NEXT J
        taskord(ii) = s
        NEXT ii
        RETURN
```